SOLUTION OF THE SINGLE BLOW PROBLEM WITH LONGITUDINAL CONDUCTION BY NUMERICAL INVERSION OF LAPLACE TRANSFORMS

FLOYD E. MORELAND

SOLUTION OF THE SINGLE BLOW PROBLEM WITH LONGITUDINAL

CONDUCTION BY

NUMERICAL INVERSION OF LAPLACE TRANSFORMS

* * * * * *

Floyd E. Moreland

SOLUTION OF THE SINGLE BLOW PROBLEM WITH LONGITUDINAL CONDUCTION BY NUMERICAL INVERSION OF LAPLACE TRANSFORMS

by

Floyd E. Moreland

Lieutenant, United States Navy

Submitted in partial fulfillment of
the requirements for the degree of

MASTER OF SCIENCE
IN
MECHANICAL ENGINEERING

United States Naval Postgraduate School
Monterey, California

1 9 6 4

SOLUTION OF THE SINGLE BLOW PROBLEM WITH LONGITUDINAL CONDUCTION BY NUMERICAL INVERSION OF LAPLACE TRANSFORMS

by

Floyd E. Moreland

This work is accepted as fulfilling

the thesis requirements for the degree of

MASTER OF SCIENCE

IN

MECHANICAL ENGINEERING

from the

United States Naval Postgraduate School

ABSTRACT

A system of two partial differential equations represents the transient heat transfer behavior of compact heat exchanger surfaces when subjected to a step change in fluid temperature. A solution is presented for this system which includes the effects of longitudinal thermal heat conduction. Also presented are the solutions for the two limiting cases of zero and infinite longitudinal conduction. The numerical results have been compared to those of C. P. Howard indicating a significant decrease in computational time and an increase in accuracy of results. The revised curves of maximum slope of fluid temperature versus NTU should be of practical value in the evaluation of heat-transfer data obtained by transient testing of compact heat exchanger surfaces. An unusual combination of mathematical techniques is presented for the solution of a boundary value problem involving partial differential equations. The solution combines the application of Laplace transformation with a numerical technique developed by H. Hurwitz, Jr., and P. F. Sweifel, and adapted by L. A. Schmittroth for the inversion of Laplace transforms. This technique greatly expands the number of cases to which Laplace transforms may be successfully applied.

ACKNOWLEDGMENTS

I would like to express my appreciation to my thesis advisor, Dr. U. R. Kodres, Associate Professor, Department of Mathematics and Mechanics, U. S. Naval Postgraduate School, for outlining the mathematical techniques applied to this solution, and for his excellent guidance in the development of the computer programs required. As a result of his active support, the computer solutions obtained are more accurate and the computation time required is considerably less.

I would also like to thank Mr. C. P. Howard, Gas Turbine Division, Code 645, Bureau of Ships, for his advice and guidance during my preliminary investigation of the single blow problem, and for the use of his computer solutions.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

NOMENCLATURE

A	total heat transfer area, ft^2.
A_s	matrix cross sectional area, ft^2.
C	step increase in fluid temperature, $= v_1 - v_o$, °F.
c_f	specific heat of fluid, BTU/lb °F.
c_s	specific heat of solid, BTU/lb °F.
h	heat transfer coefficient, BTU/hr ft^2 °F.
K_s	thermal conductivity of matrix, BTU/hr ft °F.
ℓ	total length of fluid flow path, ft.
λ	dimensionless conduction parameter, $= K_s A_s / w_f c_f$.
NTU	dimensionless heat transfer units, $= hA/w_f c_f$.
t	dimensionless time variable, $= hA\theta / W_s c_s$.
θ	time, hr.
u	solid temperature, °F.
u_o	reference solid temperature, °F.
v	fluid temperature, °F.
v_1	fluid temperature at $X = 0$, $t = 0+$; °F.
v_o	reference fluid temperature, °F.
w_f	fluid mass flow rate, lb/hr.
W_s	mass of matrix, lb.
X	distance along the fluid flow path, ft.
x	dimensionless position variable, $= X/\ell$.

1. Introduction.

The "single blow" problem refers to the study of the transient heat transfer behavior of compact heat exchanger surfaces for the purpose of determining their heat transfer characteristics. Interest in compact heat exchangers has grown with the development of gas turbine engines. A regenerative cycle is desirable to increase the efficiency of the gas turbine. Proper design of a compact heat exchanger requires knowledge of the heat transfer characteristics of the various materials in numerous geometric configurations. Until recently it has been necessary to design and build the complete heat exchanger in order to determine its heat transfer characteristics experimentally. The objectives of the single blow studies are to determine the heat transfer characteristics from small test sections of the various matrices and to obtain considerable reduction in experimental costs.

The aim of the experimental testing is to obtain the exit fluid time-temperature history subsequent to a step change in the entrance fluid temperature. The experimental history must be compared to a theoretical time-temperature history to determine NTU, the dimensionless heat transfer parameter. The heat transfer coefficient, h, or the Colburn J factor can then be calculated.

In 1950 Locke [9] outlined five experimental techniques.[1] One of the simpler of these has become known as the "maximum slope" method. This technique compares the maximum slope of the experimental time-temperature history to the theoretical maximum slopes for various values of NTU and of λ , the longitudinal conduction parameter. With a cursory inspection, one might not appreciate the efficiency of this single point curve matching

[1] Numbers in square brackets refer to the bibliography.

technique. One maximum slope represents a complete theoretical or a complete experimental curve. Therefore, it is unnecessary to calculate and plot vast quantities of theoretical time-temperature histories. This method eliminates the laborious and inaccurate matching of theoretical and experimental time-temperature histories, previously required to determine NTU. However, the maximum slope technique has two major limitations. The first problem is the difficulty in analytically solving the system of governing differential equations when longitudinal conductivities other than zero or infinity are included. The second problem is the magnification of experimental error due to curve matching. This problem is discussed in detail later.

Considerable analytical effort has been made to solve various aspects of the single blow problem since 1927 when Nusselt first studied it. Schumann [14] developed a solution in 1929 for zero conductivity in the direction of flow. His results are for a porous medium such as gravel but have been extended to include matrices consisting of metal balls, wire screens, or continuous materials for which the effects of longitudinal thermal conduction can be ignored. However, as indicated by Mondt [11] and Creswick [3], usually when the matrix is constructed of a continuous material in the flow direction, the effects of longitudinal conduction must be considered. Creswick outlined a finite difference technique to include this effect but his work was not complete enough to cover the area encountered in experimental testing. Mondt [11] obtained a closed solution for the limiting case of $\lambda = \infty$. Figure 1 indicates how drastically these two cases differ and shows the need for intermediate curves.

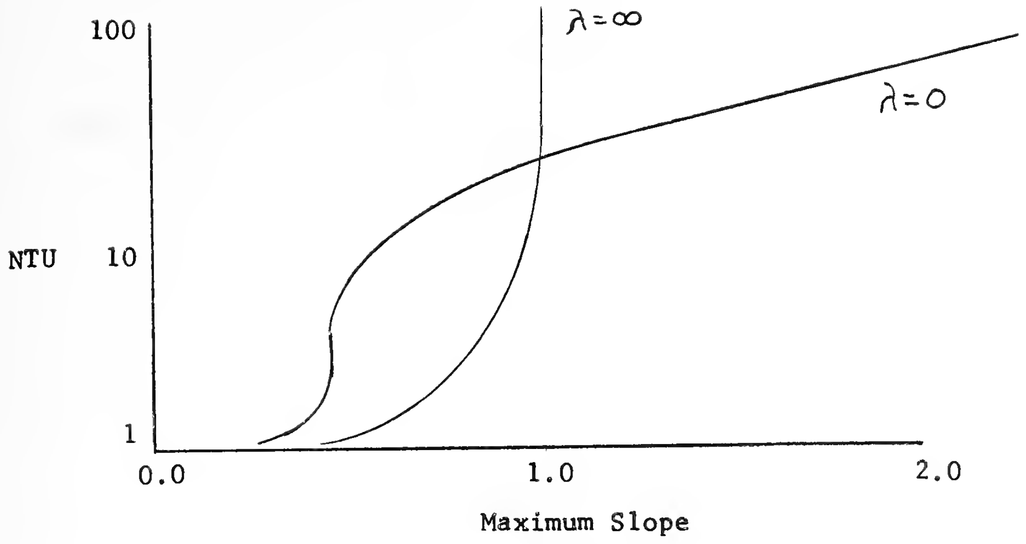


Figure 1

In 1948, F. E. Romie, et. al. [12] developed a system of partial differential equations governing the transient heat transfer behavior for hollow circular cylinders. Creswick arrived at the same set of equations for parallel plates. By a finite difference technique, he numerically approximated the solution of the system. However, difficulty with convergence was sometimes encountered. In 1963, C. P. Howard [5], guided by physical considerations, developed an alternate form of the finite-difference equations, and he was able to determine the convergence criteria and avoid the problem encountered by Creswick. Howard presented a table of results and a set of curves of NTU versus Maximum Slope for various values of λ. The table covers a range quite adequate for most experimental testing. As the values of NTU increased beyond 60, it became very difficult to obtain solutions with the finite difference technique due to the large computation times involved.

The objective of the solution presented in this paper is to verify or improve the results obtained by C. P. Howard and to avoid large computation times as NTU increases. This solution not only avoids the difficulty at

large values of NTU, but becomes faster as NTU increases due to the corresponding increase in the time at which the maximum slope occurs.

Referring to the error magnification due to curve matching, C. P. Howard [5] showed in Figure 3-A of his Appendix 3, that the error due to the experimental determination of maximum slope is magnified by a factor of two or greater when the value of NTU is determined by the maximum slope technique. The error magnification factor is defined as the ratio of percent error in NTU to percent error in experimental Maximum Slope. The essential features of the error factor as a function of NTU are indicated in Figure 2.

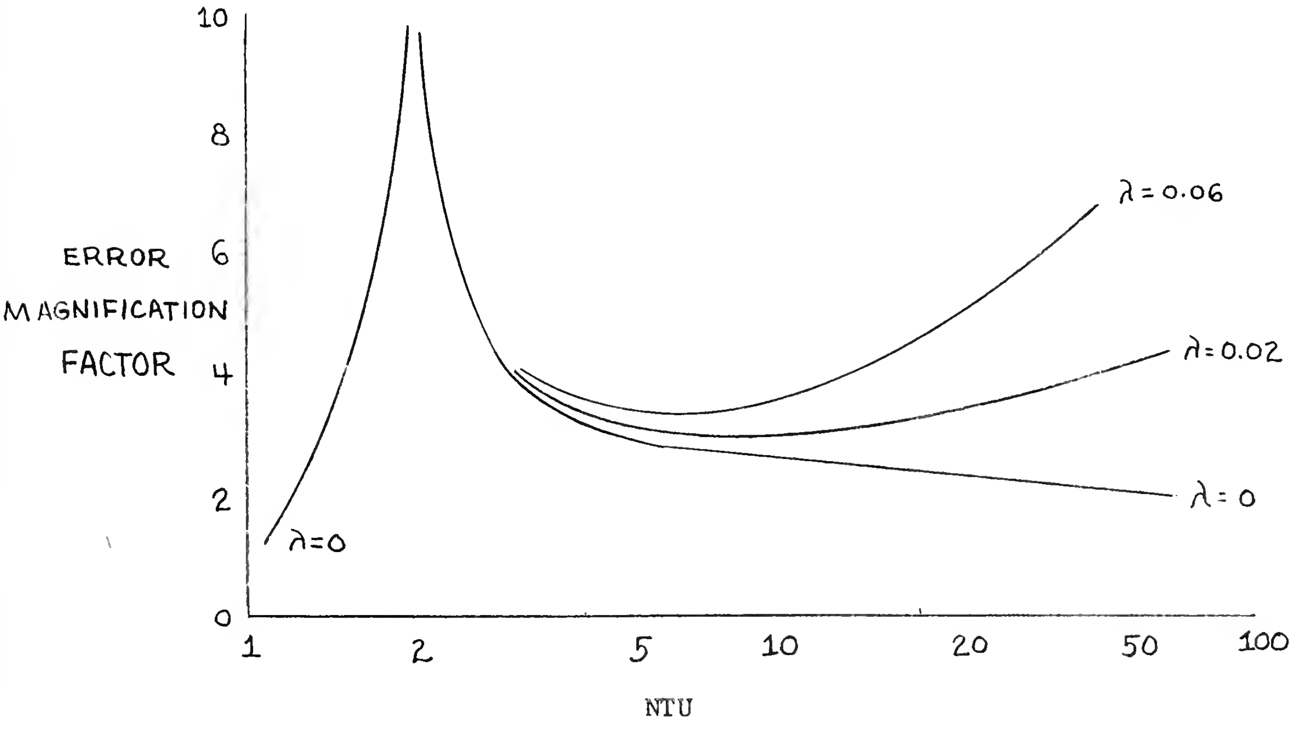


Figure 2

This figure shows that for the special case, $\lambda = 0$, the lowest error magnification is between 2.5 and 2 for NTU greater than five. At NTU near two there is a peak for which an accurate determination of NTU by maximum slope matching is impossible. When λ is non-zero, the peak

shifts to slightly higher values of NTU and the useful range above NTU of four decreases as λ increases. For example, for λ = .06, an error factor of 3, at NTU = 5, is doubled at NTU = 28. Therefore, with this technique, one is forced to accept much larger errors as NTU and λ increase. To avoid the inherent magnification of error, a modification can be made to the time-temperature curve matching technique with the aid of modern computers. When λ is finite, non-zero it is mandatory to use a computer to calculate the theoretical values of fluid temperature, given NTU, and t[2]. It would be elementary to include a least square error curve matching technique in the temperature program. Then, given three or more experimental values of fluid temperature and their corresponding values of t, the computer would compare the calculated theoretical temperatures at the given t's for various NTU's and determine the NTU with the least square error. The anticipated error in any particular theoretical value of temperature is ± 1% or less.

---

[2] "t" denotes the dimensionless time parameter.

2. Mathematical Technique.

The approach to be presented solved the system of partial differential equations for finite, non-zero $\mathcal{A}$ by eliminating the variable of solid temperature. The result is a third order partial differential equation of fluid temperature as a function of time and x (the ratio of distance along the matrix to the total length). This equation and the necessary boundary conditions were transformed with respect to time by Laplace transformations to obtain a third order total differential equation with respect to x with parameter s. The equation was then solved for the transformed fluid temperature. The transformed boundary conditions were applied to determine the three arbitrary coefficients which were functions of the parameter s. Because the transformed solution was very complicated, numerical inversion was used to compute the inverse Laplace transform.

A Gaussian quadrature method, developed by H. Hurwitz, Jr. and P. F. Sweifel [6] for Fourier Transform Integrals, was adapted by L. A. Schmittroth [13] for the numerical inversion of Laplace Transforms. This technique avoids the difficulty encountered with alternating, slowly converging functions and proved to be essential for the successful inversion of this solution. For the two limiting cases $\mathcal{A} = 0$ and $\mathcal{A} = \infty$, a direct inversion of the Laplace solution was available.

3. Solution.

The governing differential equations for the transient heat transfer behavior including a finite, non-negative, longitudinal conductivity are as follows:[3]

$$(1)\qquad \frac{\partial u}{\partial t} = (v-u) + \frac{\lambda}{NTU}\frac{\partial^2 u}{\partial x^2}$$

$$(2)\qquad \frac{\partial v}{\partial x} = NTU\,(u-v)$$

The applicable boundary conditions are the following:

a. $V(x,0) = 0$

b. $\frac{\partial v}{\partial x}(x,0) = 0$

c. $v(0,t) = C$

d. $\frac{\partial u}{\partial x}(o,t) = 0$

e. $\frac{\partial u}{\partial x}(1,t) = 0$

The following is the approach used to solve this set of equations for the fluid temperature, v, and the slope of fluid temperature, $\frac{\partial v}{\partial t}$, respectively. In a similar manner the solid temperature, u, and other desired values can be obtained.

Solving equation 2 for u results in

$$(2a)\qquad u = \frac{1}{NTU}\frac{\partial v}{\partial x} + v$$

The partial derivative of u with respect to t and the second partial of u with respect to x are determined from 2a, and then substituted into equation 1 to give

$$(3)\qquad 0 = \frac{\partial^3 v(x,t)}{\partial x^3} + NTU\frac{\partial^2 v(x,t)}{\partial x^2} - \frac{NTU}{\lambda}\frac{\partial v(x,t)}{\partial x} - \frac{NTU^2}{\lambda}\frac{\partial v(x,t)}{\partial t} - \frac{NTU}{\lambda}\frac{\partial^2 v(x,t)}{\partial x\,\partial t}$$

[3]The equations were developed from a heat balance by Creswick [3] .

Each term in equation 3 is transformed with respect to t by Laplace transformation along with the following boundary conditions:

a. $v(x,0) = 0$

b. $\frac{\partial v}{\partial X}(x,0) = 0$

The necessary transforms are

c. $\mathcal{L}\left\{\frac{\partial^3 v}{\partial X^3}\right\} = \frac{\partial^3 v}{\partial X^3}(X,S)$ etc.,

d. $\mathcal{L}\left\{\frac{\partial v}{\partial t}(x,t)\right\} = \int_0^\infty \frac{\partial v}{\partial t} e^{-st} dt = v(x,t)e^{-st}\Big|_0^\infty + \int_0^\infty S\,[v(x,t)]e^{-st} dt$

$= -v(X,0) + S[v(X,S)]$

e. $\mathcal{L}\left\{\frac{\partial^2 v(x,t)}{\partial x \partial t}\right\} = -\frac{\partial v}{\partial X}(X,0) + S\frac{\partial v}{\partial X}(X,S)$

Substitution of these values in equation 3 after applying the above boundary conditions, results in

(3a). $0 = \frac{\partial^3 v(X,S)}{\partial X^3} + NTU\frac{\partial^2 v(X,S)}{\partial X^2} - \frac{NTU(S+1)}{R}\frac{\partial v(X,S)}{\partial X} - \frac{NTU^2 S}{R} v(X,S)$.

The corresponding auxiliary equation is

(3b). $r^3 + NTU\, r^2 - \frac{NTU}{R}(S+1)r - \frac{NTU^2}{R}S = 0$

The general solution in the Laplace s-plane for the fluid temperature is

(4). $v(X,S) = C_1(S)e^{r_1 X} + C_2(S)e^{r_2 X} + C_3(S)e^{r_3 X}$

where r1, r2, r3 are the roots of equation (3b).

The boundary conditions c, d, and e are transformed and then used to determine the coefficients C1, C2, and C3. The transform of C is

$v(0,S) = \frac{C}{S}$.

The transform of d is $\frac{\partial u}{\partial X}(0,S) = 0$, and similarly for e, $\frac{\partial u}{\partial X}(1,s) = 0$.

From equation 2a the following equation is obtained:

$$(5).\quad \frac{\partial u}{\partial X}(X,t) = \frac{1}{NTU}\frac{\partial^2 v}{\partial X^2} + \frac{\partial v}{\partial X}(X,t).$$

Applying boundary condition c to equation 4 gives

$$(6).\quad v(0,s) = C_1(s) + C_2(s) + C_3(s) = \frac{\mathcal{C}}{s}.$$

Rather than apply boundary conditions d and e directly it is convenient to use their equivalent form in terms of v. To apply boundary condition d it is necessary to transform equation 5 which results in

$$\frac{\partial u}{\partial X}(X,s) = \frac{1}{NTU}\frac{\partial^2 v}{\partial X^2}(X,s) + \frac{\partial v}{\partial X}(X,S) \qquad \text{therefore;}$$

$$\frac{\partial u}{\partial X}(0,S) = \frac{1}{NTU}\frac{\partial^2 v(0,s)}{\partial X^2} + \frac{\partial v}{\partial X}(0,S) = 0.$$

When this boundary condition is applied the following equation is obtained:

$$(7).\quad \frac{1}{NTU}\left(r_1^2 C_1(s) + r_2^2 C_2(s) + r_3^2 C_3(s)\right) + r_1 C_1(s) + r_2 C_2(s) + r_3 C_3(s) = 0.$$

The application of boundary condition e is similar, resulting in

$$(8).\quad \frac{1}{NTU}\left(r_1^2 C_1 e^{r_1} + r_2^2 C_2 e^{r_2} + r_3^2 C_3 e^{r_3}\right) + r_1 C_1 e^{r_1} + r_2 C_2 e^{r_2} + r_3 C_3 e^{r_3} = 0.$$

(8a). Define $R_m \equiv \left(\frac{r_m^2}{NTU} + r_m\right)$ where n = 1, 2, 3. Then the following matrix equation can be set up using equations 6, 7, 8.

$$\begin{bmatrix} \frac{C}{s} \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ R_1 & R_2 & R_3 \\ R_1 e^{r_1} & R_2 e^{r_2} & R_3 e^{r_3} \end{bmatrix} \cdot \begin{bmatrix} C_1(s) \\ C_2(s) \\ C_3(s) \end{bmatrix}$$

To solve for C1, C2, C3, Cramer's Rule is applied. The denominator is

$$\Delta = \left[R_2 R_3 (e^{r_3} - e^{r_2}) + R_1 R_3 (e^{r_1} - e^{r_3}) + R_1 R_2 (e^{r_2} - e^{r_1}) \right].$$

The three arbitrary coefficients are the following:

$$C_1(s) = \frac{\begin{vmatrix} C/s & 1 & 1 \\ 0 & R_2 & R_3 \\ 0 & R_2 e^{r_2} & R_3 e^{r_3} \end{vmatrix}}{\Delta} = \frac{\frac{C}{s}\left[R_2 R_3 e^{r_3} - R_2 R_3 e^{r_2} \right]}{\Delta},$$

$$C_2(s) = \frac{\begin{vmatrix} 1 & C/s & 1 \\ R_1 & 0 & R_3 \\ R_1 e^{r_1} & 0 & R_3 e^{r_3} \end{vmatrix}}{\Delta} = \frac{\frac{C}{s}\left[R_1 R_3 e^{r_1} - R_1 R_3 e^{r_3} \right]}{\Delta}$$

and

$$C_3(s) = \frac{\begin{vmatrix} 1 & 1 & C/s \\ R_1 & R_2 & 0 \\ R_1 e^{r_1} & R_2 e^{r_2} & 0 \end{vmatrix}}{\Delta} = \frac{\frac{C}{s}\left[R_1 R_2 e^{r_2} - R_1 R_2 e^{r_1} \right]}{\Delta}.$$

Substituting C1, C2, and C3 into equation 4 results in

$$v(x,s) = \frac{C}{s}\,\frac{\left[R_2R_3(e^{r_3}-e^{r_2})e^{r_1x}+R_1R_3(e^{r_1}-e^{r_3})e^{r_2x}+R_1R_2(e^{r_2}-e^{r_1})e^{r_3x}\right]}{\Delta} \tag{9}$$

Evaluating equation 9 at x = 1 results in the following:

$$v(1,s) = \frac{C}{s}\,\frac{\left\{R_2R_3\left[e^{(r_3+r_1)}-e^{(r_2+r_1)}\right]+R_1R_3\left[e^{(r_1+r_2)}-e^{(r_3+r_2)}\right]+R_1R_2\left[e^{(r_2+r_3)}-e^{(r_1+r_3)}\right]\right\}}{R_2R_3(e^{r_3}-e^{r_2})+R_1R_3(e^{r_1}-e^{r_3})+R_1R_2(e^{r_2}-e^{r_1})} \tag{10}$$

Recall that the Laplace operation $Sf(s) - F(+0)$ corresponds to $\frac{d}{dt}F(t)$. It follows directly that $\frac{\partial v}{\partial t}(1,t)$ is equal to the operation $s \cdot v(1,s)$, since $v(1,0) = 0$ by boundary condition a. Therefore,

$$\frac{\partial v}{\partial t}(1,s) = s \cdot v(1,s) \tag{11}$$

To determine the exit fluid temperature and the slope with respect to time of the exit fluid temperature, it is necessary to find the inverse Laplace transforms of equations 9 and 10 respectively. If the values of R_i and r_i, i = 1, 2, 3; were not functions of s this step would be elementary. Unfortunately, the values of R and r are indeed functions of s as indicated by equations 8a and 3b respectively. Therefore, for this particular case the task of finding an analytic inverse transform is nearly hopeless.

Rather than proceed further on this approach, a computer program to calculate the complex roots for given values of s, NTU, and λ, is used. Consequently, this step precludes any possibility of finding an analytical solution if one exists. Now, the value of v(1,s) or $\frac{\partial v}{\partial t}$ at (1,s) can be determined for any set of the parameters, C, s, NTU, and λ,

and the Laplace inversion integral may be used to find v(1,t). Finding an adequate numerical technique to evaluate this inversion integral was no simple matter.

When Simpson's Rule was applied to evaluate the inversion integral using the real (VR) and imaginary (VI) parts of v(1,s), the results were grossly in error, the behavior being as described in reference [6] and [13]. The numerical technique described in reference [13] was found to be greatly superior to Simpson's Rule for this particular problem. It incorporates a technique to accelerate convergence of an alternating series, and uses Gauss-Chebyshev formulas for integration. This technique uses either the real part (VR) or the imaginary part (VI) and results in two different forms. Since there are no known values for this solution it was necessary to use both of these forms to be confident that the correct solutions were obtained.

According to the theory of Laplace inversion, the inversion integral,[4] $\frac{1}{2\pi i}\int_{G-i\infty}^{G+i\infty} v(x,s)\, e^{st}\, ds$, should be independent of G, where $s = G + iy$. When this approach was used with Simpson's Rule none of the results were independent of G. Recall the form of the inversion integral $\frac{e^{Gt}}{\pi}\int_{0}^{i\infty} v(x,G,y)\, e^{iyt}\, dy$. Since the region initially investigated used t = 7 and G of .5, 1 and 2, any error in the numerical technique was greatly multiplied giving very poor results, However; this indicated that G is an additional parameter that must be determined for each new program. Rather than cause difficulty, this parameter proves to be essential. For the same G, it was found that one program gave consistently low results while the other was consistently high. Then by

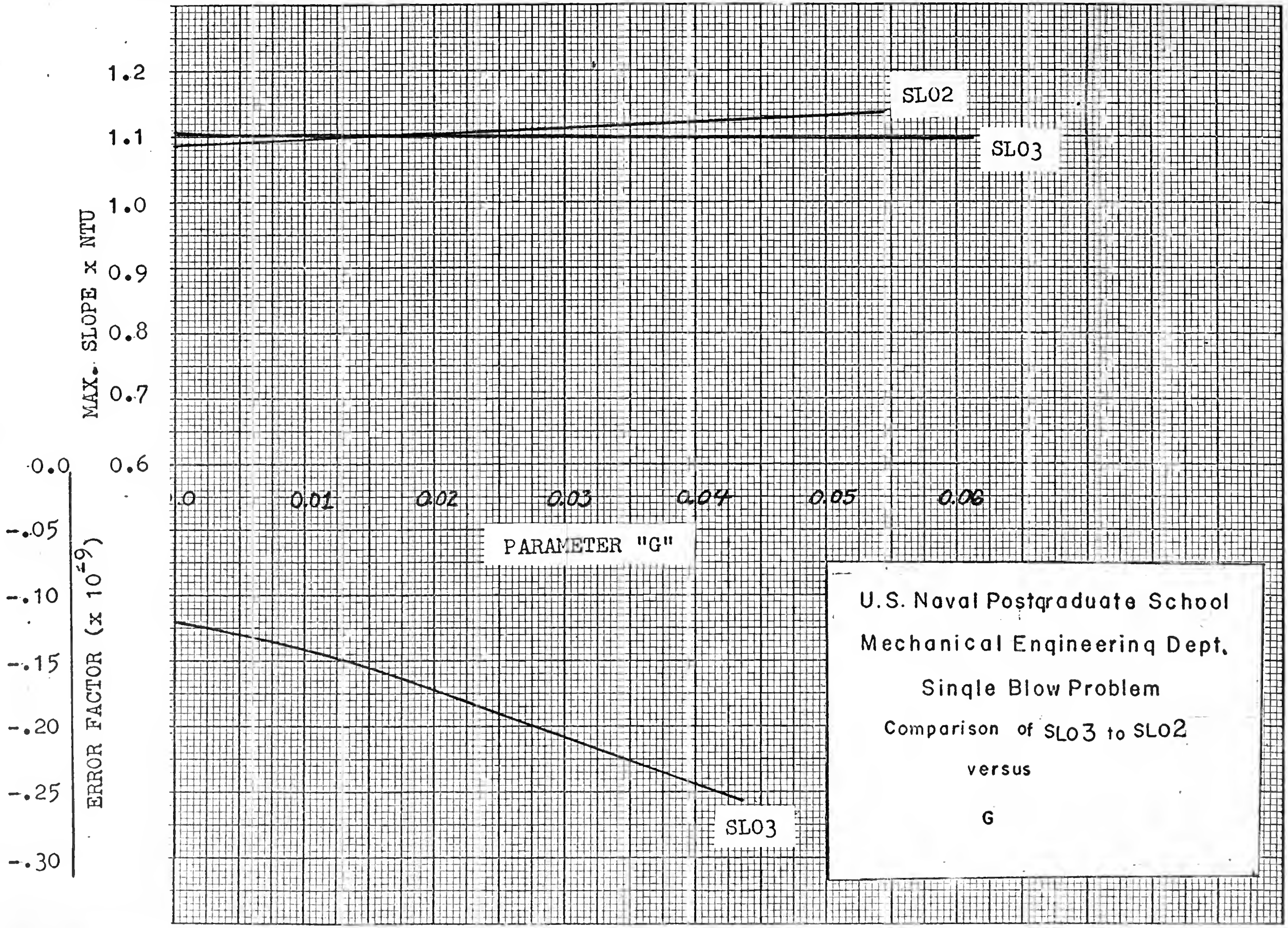
[4]Churchill, op. cit., p. 176.

varying G of both programs the program using VR can be made to agree with the results of the program using VI. This is the approach that was used to make the various programs converge to a solution. The numerical technique outlined in [13] uses Chebyshev's polynomial to fit the function f(s). It was found that this polynomial is very sensitive to a small change in G. For example, upon changing G from 0.0 to 0.05, the product (slope • NTU) changed from 1.0877 to 1.1333 an increase of 4.19%. Again according to theory, results should be independent of G. Therefore, two new programs were written using a Gauss-Legendre quadrature format to integrate VR or VI. These programs were found to be essentially independent of G for reasonable increments of G. For example changing G from 0.0 to 0.06, the slope•NTU changed from 1.10096 to 1.09776 a decrease of 0.29%. Figure 3 indicates a comparison of these two programs. On this basis alone, it is felt that the Legendre polynomial is more accurate. Unfortunately, in comparison to the programs using the Chebyshev polynomial it is considerably slower. Because of this fact, the Legendre programs which include an error analysis are being used to determine accurate test values for the determination of the best G for each program. Once the best G is determined, the speedier Chebyshev programs can be used for long data runs.

The integration technique described in detail in references [6] and [13] will be described here as it was applied to this problem. The inversion integral is the following:

$$\mathcal{L}^{-1}\{f(s)\} = f(t) = \frac{1}{2\pi i}\int_{G-i\infty}^{G+i\infty} f(s)\, e^{-st}\, ds \tag{12}$$

Then writing f(s) as the sum of the real and imaginary parts results in



U.S. Naval Postgraduate School
Mechanical Engineering Dept.
Single Blow Problem
Comparison of SL03 to SL02
versus
G
MAX. SLOPE x NTU
1.2
1.1
1.0
0.9
0.8
0.7
0.6
ERROR FACTOR (x 10^-9)
0.0
-.05
-.10
-.15
-.20
-.25
-.30
PARAMETER "G"
0.0
0.01
0.02
0.03
0.04
0.05
0.06
SL02
SL03
SL03

FIGURE 3

$$f(t) = \frac{1}{2\pi i}\int_{-\infty}^{\infty}[\phi(G+iy) + i\psi(G+iy)]\, e^{t(G+iy)}\,(i\,dy).$$

The inversion integral can then be written as

$$(12a)\quad f(t) = \frac{e^{Gt}}{2\pi}\left\{\int_{-\infty}^{0}[\phi(G+iy)+i\psi(G+iy)]e^{iyt}dy + \int_{0}^{\infty}[\phi(G+iy)+i\psi(G+iy)]e^{iyt}dy\right\},$$ [5]

where the following condition applies: $f(\bar{s}) = \overline{f(s)}$.

It follows directly that

$$(12b)\quad \phi(G+iy) = \phi(G-iy)\,,\quad \text{and}$$

$$(12c)\quad \psi(G+iy) = -\psi(G-iy).$$

In the first integral of equation 12a, let $y = -y'$ and $dy = -dy'$, then

$$(13)\quad f(t) = \frac{e^{Gt}}{2\pi}\left\{\int_{0}^{\infty}[\phi(G-iy')+i\psi(G-iy')]e^{-iy't}dy' + \int_{0}^{\infty}[\phi(G+iy)+i\psi(G+iy)]e^{iyt}dy\right\}.$$

Applying the relations 12b and 12c, equation 13 reduces to

$$f(t) = \frac{e^{Gt}}{2\pi}\int_{0}^{\infty}\left\{[\phi(G+iy)-i\psi(G+iy)]e^{-iyt}+[\phi(G+iy)+i\psi(G+iy)]e^{iyt}\right\}dy.$$

On rearranging and simplifying, this becomes

$$(14)\quad f(t) = \frac{e^{Gt}}{\pi}\int_{0}^{\infty}[\phi(G+iy)\cos yt - \psi(G+iy)\sin yt]\,dy.$$

By definition of the inverse integral, $f(-t) = 0$. Thus

$$(15)\quad 0 = \frac{e^{-Gt}}{\pi}\int_{0}^{\infty}[\phi(G+iy)\cos yt + \psi(G+iy)\sin yt]\,dy,$$

[5] A bar over a quantity indicates the complex conjugate.

which reduces to

$$\int_0^{\infty} \phi(G+iy)\cos yt\,dy = -\int_0^{\infty} \psi(G+iy)\sin yt\,dy.$$

Two forms of f(t) can be found by expressing the integrals either in terms of the real or the imaginary parts. They are respectively

(16) $$f(t) = \frac{2e^{Gt}}{\pi}\int_0^{\infty} \phi(G+iy)\cos yt\,dy,$$ and

(17) $$f(t) = \frac{-2e^{Gt}}{\pi}\int_0^{\infty} \psi(G+iy)\sin yt\,dy.$$

These two equations are the basis of the two programs mentioned earlier. In general discussions VR will represent the function ϕ cos yt and VI the function ψ sin yt. The Gaussian quadrature formula using either Chebyshev's or Legendre's polynomial is used to evaluate the integral from zero to infinity. Both functions VR and VI are of the general form shown in Figure 4

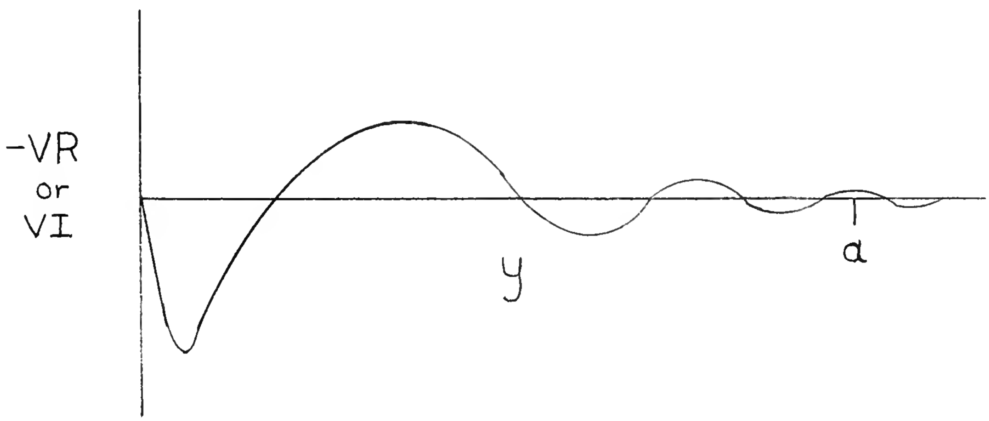


Figure 4

If the function were integrated with respect to y and the summation stopped at a, there would be significant error. Two operations are

included in this solution to reduce the error. The first operation is an averaging technique which accelerates convergence. The second operation involves integrating each half wave and then summing them. To accomplish this we introduce the following change of variable in Equation 17.

Let $x = \frac{t}{\pi} y - \frac{1}{2}$ therefore $y = \frac{\pi}{t}(x + \frac{1}{2})$ and $dy = \frac{\pi}{t} dx$.

Applying these to equation 17 results in

$$\text{(17a)} \quad f(t) = -\frac{2e^{Gt}}{t} \int_{-1/2}^{\infty} \psi\left[G + i\frac{\pi}{t}(x + \tfrac{1}{2})\right] \sin\pi(x + 1/2)\, dx,$$

$$= -\frac{2e^{Gt}}{t} \sum_{m=0}^{\infty} \int_{-\frac{1}{2}+m}^{\frac{1}{2}+m} \psi\left[G + i\frac{\pi}{t}(x + \tfrac{1}{2})\right] \sin\pi(x + \tfrac{1}{2})\, dx.$$

Now let $x' = x - m$, then

$$\text{(17b)} \quad f(t) = -\frac{2e^{Gt}}{t} \sum_{m=0}^{\infty} \int_{-1/2}^{1/2} \psi\left[G + \frac{i\pi}{t}(x' + m + \tfrac{1}{2})\right] \sin\pi(x' + m + \tfrac{1}{2})\, dx'.$$

Upon supressing the prime, we have

$$\text{(18)} \quad f(t) = \frac{2e^{Gt}}{t} \sum_{m=0}^{\infty} (-1)^{m+1} \int_{-\frac{1}{2}}^{\frac{1}{2}} \psi\left[G + i\frac{\pi}{t}(x + m + \tfrac{1}{2})\right] \cos\pi x\, dx$$

When equation 18 is written in terms of partial sums, the result is

$$\text{(19)} \quad f(t) = \frac{2e^{Gt}}{t} \sum_{m=0}^{\infty} I_m(t)$$

$$= \frac{2e^{Gt}}{t} \lim_{m \to \infty} S_m(t), \quad \text{where}$$

$$S_m(t) = \sum_{m=0}^{m} I_m(t) \quad \text{and}$$

$$(19a)\quad I_m(t) = (-1)^{m+1} \int_{-\frac{1}{2}}^{\frac{1}{2}} \Psi\left[G + i\frac{\pi}{t}(x+m+\tfrac{1}{2})\right] \cos(\pi x)\, dx .$$

Again referring to [13], L. A. Schmittroth applied a Gaussian quadrature method to evaluate $I_n(t)$. $I_n(t)$ can be approximated as follows:

$$(19b)\quad I_m(t) \cong (-1)^{m+1} \sum_{j=1}^{N} \frac{W_j^N}{\cos \pi y_j^N} \left[\Psi(-\omega) + \Psi(+\omega)\right]$$

where

$$(19c)\quad y_j^N = \frac{2j-1}{2(2N+1)} \quad , \; j = 1, 2, 3, \cdots, N$$

and $-\omega = \frac{\pi}{t}\left(-y_j^N + m + \frac{1}{2}\right)$; $+\omega = \frac{\pi}{t}\left(y_j^N + m + \frac{1}{2}\right)$.

The number of points used to fit the function is 2 N. Obviously, the larger N is, the greater the degree of accuracy one can expect. This is the core of the iterative process and an adequate N is desired to ensure reasonable computer time. This point will be discussed in detail later. The N coefficients W_j^N are the solutions of the system.

$$(19d)\quad 2 \sum_{j=\lambda}^{N} W_j^N \cos^{2N-2}\left[\frac{(2j-1)\pi}{2(2N+1)}\right] = \frac{1}{\sqrt{\pi}} \frac{\Gamma(p+\frac{1}{2})}{\Gamma(p+1)} ,$$

where $p = 1, 2, \ldots, N$.

Equation 19 uses Chebyshev's polynomial for the curve fitting. A similar form of $I_n(t)$ can be developed from Equation 16. As mentioned earlier, an integration technique was developed using a Legendre polynomial for curve fitting. If equation 16 is used to illustrate this approach, the relation $\lambda = \frac{ty}{\pi}$ is required. Then equation 16 reduces to:

$$(20)\quad f(t) = \frac{2e^{Gt}}{t}\int_0^{\infty} \phi\left(G + i\frac{\pi}{t}x\right) \cos \pi x \, dx.$$

Now, the Gaussian quadrature method can be applied as above. For the Gauss-Legendre formulas, a change of variable was applied to equation 17b to set up the infinite sum of integrals from zero to one, one to two and so on rather than a sum of integrals evaluated from -1/2 to 1/2.

The Gauss-Chebyshev and the Gauss-Legendre formulas applied to the real or imaginary form of the inversion integral produce four different computer programs. Any one of these should give the correct solution, but one form may prove to be better for a particular application.

For zero and infinite longitudnal conduction, the governing differential equations reduce to much simpler forms. The solutions for both cases are given in Appendix I and are in complete agreement with the solutions of Schumman and Mondt for $\lambda = 0$ and $\lambda = \infty$ respectively.

4. Computer Program.

The principal objective of the computer program is to solve for the maximum slope[6] of fluid temperature at $X = \ell$. The main program is designed to determine the slope of the fluid temperature at $X = \ell$ which is the numerical inversion of equation 11.

The subroutine SOOTS2 calculates the real and imaginary coefficients of the equation 3b given the parameters NTU, λ, G, and Y. These coefficients are then put into subroutine TOOTS2.

Subroutine TOOTS2 calculates the real and imaginary parts of the roots r_1, r_2, r_3 of equation 3b. It sends its results back to SOOTS2 which arranges them in the proper order and subscripts them appropriately. SOOTS2 then provides the proper roots as input values to EVAL. The EVAL subroutine provides the main program with the values of the real (VR) and imaginary part (VI) of $\frac{\partial v}{\partial t}(1, S)$, given the parameters G, $\frac{\pi}{t}X$, C, and the roots from SOOTS2.

The main program will call for N of these values (see equation 19c) with which it will calculate $I_o(t)$ (see equation 19). The programmer is free to determine the number of $I_n(t)$'s to sum, where $\sum_{n=0}^{m} I_n(t) = S_m(t)$. Fifteen to twenty partial sums appear to be sufficient for the function encountered in this problem. Each partial sum calculated is stored and serves as an input to subroutine AVER.

The purpose of subroutine AVER is to accelerate convergence of the partial sums by an averaging technique. This technique is discussed in reference [6]. The subroutine will take an arbitrary number of partial sums and calculate any specified n^{th} average, and return it to the main program. It also calculates the difference between successive n^{th} averages

[6] i.e., $\frac{\partial v}{\partial t}$.

returning it to the main program. The main program then requires this difference to be less than an arbitrarily specified value (EP). If the difference is larger than EP, the main program increases the number of partial sums by LD and calls AVER again until the difference is less than EP. At this point, the value of the accelerated partial sum is accepted and the slope is calculated. However, the slope calculated for a particular value of dimensionless time is not the value desired. It is necessary to search for the maximum slope by varying time.

Initially, a search using an incremental step of time was used. This was eventually abandoned because it would find relative maxima only and it was tediously slow if a poor choice were made for a starting time. It should be noted that theoretical relative maxima have been found for small values of NTU and negative slopes have also been noted. J. M. Bannon[1] is concurrently conducting an experimental thesis and has experimentally found relative maxima and negative slopes for high mass flow or small values of NTU. It would be interesting to compare a theoretical time temperature curve for his experimental value of λ with his experimental time-temperature curve.

The purpose of SLOMAX is to find the maximum slope. To avoid the problem of choosing a poor starting time, a reasonable range of t is chosen. A rule of thumb is that the largest maximum slope will occur at a time less than or equal to NTU. SLOMAX uses a parabola curve fitting technique with three points.

The first and last of the three points are program input values of time and the middle point is the arithmetic mean. The slopes are calculated for the first two values of time and then tested to ensure that the

middle slope is larger than the first. This is necessary to obtain the maximum rather than a minimum slope. Then the third value of slope is calculated and tested to insure that it is less than the middle slope. The program has the ability to adjust the arbitrary values of time to ensure the middle slope is the largest. Once this basic shape is obtained the program calculates the value of time at which the parabola has a maximum. This process is repeated using the calculated maximum time and slope as the middle point and the two nearest points from the preceeding iteration. When the difference of the new value of maximum slope and the previous calculated maximum slope is less than EP1, a program input, the slope is accepted as the maximum. Then the values of time, slope and input parameters are printed out.

TABLE 1. INDEPENDENCE OF SL03 AND SL05 FROM G

PROGRAM	G	NTU	SLOPE x NTU	TIME AT MAX. SLOPE
SL05	0.0	5	.68803355	3.39803
SL05	0.1	5	.68803300	3.39796
SL03	-0.1	10	.92857109	8.45713
SL03	0.0	10	.92857102	8.45713
SL03	0.1	10	.92857099	8.45713
SL05	-0.1	10	.92857093	8.45713
SL05	0.0	10	.92857086	8.45713
SL05	0.1	10	.92857097	8.45713
SL03	0.0	20	1.28623734	18.48003
SL03	0.1	20	1.28623734	18.48003
SL05	0.0	50	2.00992025	48.49231
SL05	0.1	50	2.00992025	48.49231
SL03	0.0	100	2.8316134	98.49618
SL03	0.1	100	2.8316134	98.49610

TEST FOR SUFFICIENT NUMBER OF PARTIAL SUMS

PROGRAM	PARTIAL SUMS	SLOPE x NTU	TIME AT MAX. SLOPE	NTU
SL03	25	.68803060	3.39793	5
SL03	95	.68803060	3.39793	5
SL05	20	1.28623734	18.47998	20
SL05	50	1.28623734	18.47999	20
SL05	1000	1.28623734	18.48001	20
SL03	20	2.00992025	48.49231	50
SL03	50	2.00992025	48.49234	50
SL03	100	2.00992025	48.49234	50

TABLE 2. RESULTS OF SL03 AND SL05 FOR λ = 0.0

NTU	PROGRAM	SLOPE x NTU	TIME AT MAX.SLOPE	R.E.MAXIM SLOPE x NTU	TIME AT MAX. SLOPE(MAXIM)
1	SL05	.36845539	.12479	.367878	.00001
1.5	SL05	.52618194O	.15336	.502041	.000015
2	SL05	.60663514	.18240	.541341	.00002
3.	SL05	.57699186	1.25824	.576594	1.272
	SL03	.57634836	1.27800		
5	SL05	.68803355	3.39803	.688032	3.40
	SL03	.68803060	3.39793		
10	SL05	.92857093	8.45713	.928571	8.46
	SL03	.92857102	8.45713		
20	SL05	1.28623734	18.47998	1.286237	18.48
	SL03	1.28623734	18.48003		
50	SL05	2.00992025	48.49234	2.009920	48.50
	SL03	2.00992025	48.49231		
100	SL05	2.8316134	98.49625	—	—
	SL03	2.8316134	98.49618		
	COMPUTATION TIME				
TIME < 10	SL05	46.0 seconds/search			
TIME > 10	SL05	25.9 seconds/search			
	SL03	31.3 seconds/search			

12ND NPS 95 (11-59)

TABLE 3. INDEPENDENCE OF SL05 FROM G FOR λ = .04

PROGRAM	G	NTU	SLOPE x NTU	TIME AT MAX. SLOPE
SL05	0.0	10	.89545795	7.43133
SL05	0.1	10	.89545744	7.43132

COMPARISON OF SL03 AND SL05

PROGRAM	G	NTU	SLOPE x NTU	TIME AT MAX. SLOPE
SL03	0.0	10	.89545690	7.43140
SL05	0.0	10	.89545795	7.43133

TEST FOR SUFFICIENT NUMBER OF PARTIAL SUMS

PROGRAM	PARTIAL SUMS	SLOPE x NTU	TIME AT MAX. SLOPE	NTU
SL05	20	.89545795	7.43133	10
SL05	50	.89545795	7.43138	10
SL05	100	.89545795	7.43136	10

RESULTS OF SL05 FOR λ = 0.04

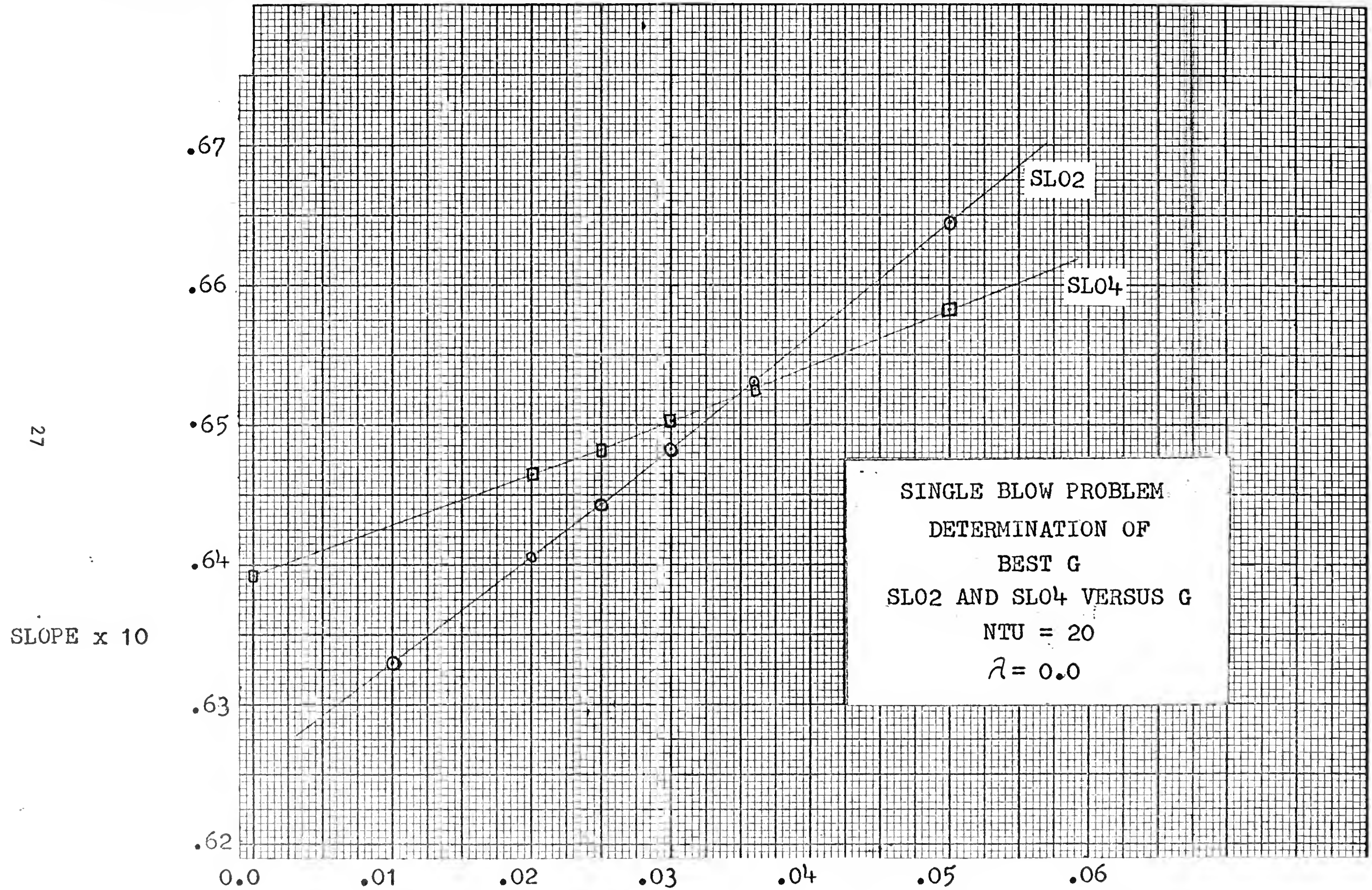
NTU	PROGRAM	SLOPE x NTU	TIME AT MAX.SLOPE	CPH SOLUTION	TIME AT MAX,SLOPE	% DIFF.OF CPH COMPARED TO SL05
2	SL05	.60485	.18037	.548	——	-9.42
5	SL05	.70586	3.05083	.704	——	-0.26
10	SL05	.89546	7.43133	.88926	7.404	-0.69
20	SL05	1.09744	16.18859	1.08959	16.164	-0.72
50	SL05	1.31977	42.61899	1.31301	42.696	-0.51
100	SL05	1.42976	87.02667	——	——	——

COMPUTATION TIME

8 to 15 minutes/search

TABLE 4. RESULTS OF SL02 AND SL04 FOR $\lambda = 0.0$

NTU=10/G	PROGRAM	SLOPE x NTU	GRAPHICAL BEST G	GRAPHICAL SLOPE x NTU	R. MAXIM SOLUTION	% DIFFF. OF SL02 & SL04 COMPARED TO MAXIM
G						
.056	SL04	.93855415				
.056	SL02	.93823756				
	mean	.93839585	.0585	.9389	.929	+1.07
NTU = 20						
G						
.036	SL04	1.30535030				
.036	SL02	1.30625902				
	mean	1.30580966	.03475	1.3042	1.286	+1.38
NTU = 50						
G						
.0125	SL04	2.02431970				
.0125	SL02	2.02571720				
	mean	2.02501895	.01175	2.0195	2.010	+0.47
NTU =100						
G						
0.0	SL04	2.7173475				
.0125	SL02	3.0172562				
	mean	2.8673018	.0055	2.825	—	—
NTU = 5						
G						
0.0	SL04	.69081005				
.03	SL02	.69395495				
	mean	.6923825	-.045	.6905	.688	+0.36
NTU = 2						
G						
0.0	SL04	.5226				
0.0	SL02	.5078				
	mean	.5152	—	—	.541	-4.80



SINGLE BLOW PROBLEM
DETERMINATION OF
BEST G
SL02 AND SL04 VERSUS G
NTU = 20
$\lambda = 0.0$
SL02
SL04
SLOPE x 10
.67
.66
.65
.64
.63
.62
0.0
.01
.02
.03
.04
.05
.06
G

FIGURE 5

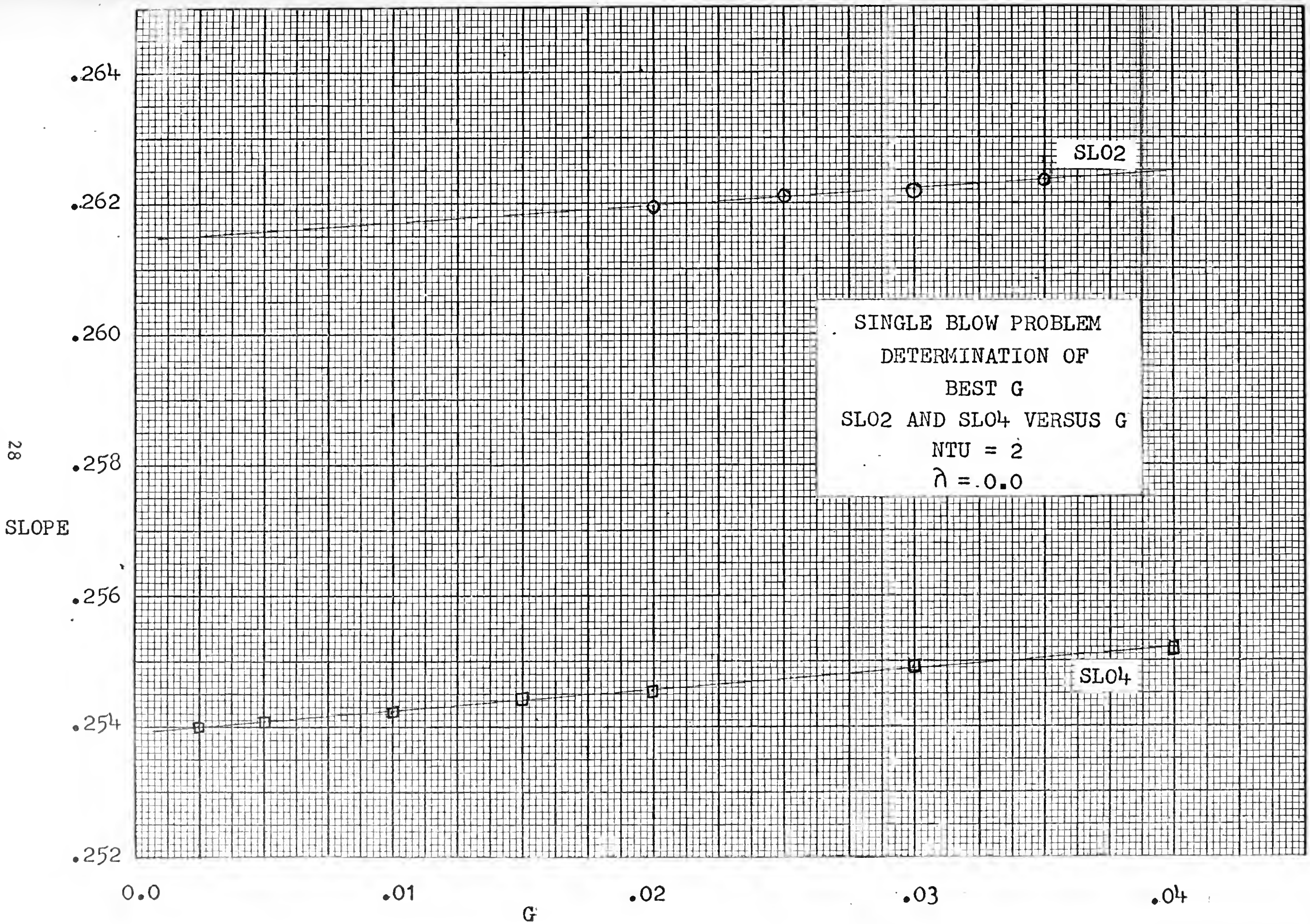

SINGLE BLOW PROBLEM
DETERMINATION OF
BEST G
SL02 AND SL04 VERSUS G
NTU = 2
λ = 0.0
SL02
SL04
.264
.262
.260
.258
.256
.254
.252
SLOPE
0.0
.01
.02
.03
.04
G

FIGURE 6

TABLE 5. RESULTS OF SLO2 AND SLO4 FOR λ = 0.005

NTU=10/G	PROGRAM	SLOPE x NTU	GRAPHICAL BEST G	GRAPHICAL SLOPE x NTU	CPH SOLUTION	% DIFF. OF CPH COMPARED TO SLO 2 & SLO4
G						
.056	SLO4	.93328101	.056	.93329	.922	-1.20%
NTU = 20						
G						
.0365	SLO4	1.27202790				
.0365	SLO2	1.27155066				
	mean	1.27178928	.0365	1.2716	1.249	-1.81%
NTU = 50						
G						
.015	SLO4	1.86727320				
.015	SLO2	1.86971140				
	mean	1.86849230	.0144	1.8643	1.833	-1.68%
NTU =100						
G						
.01	SLO4	2.4381850				
.01	SLO2	2.4568748				
	mean	2.4475299	.0067	2.386	—	—

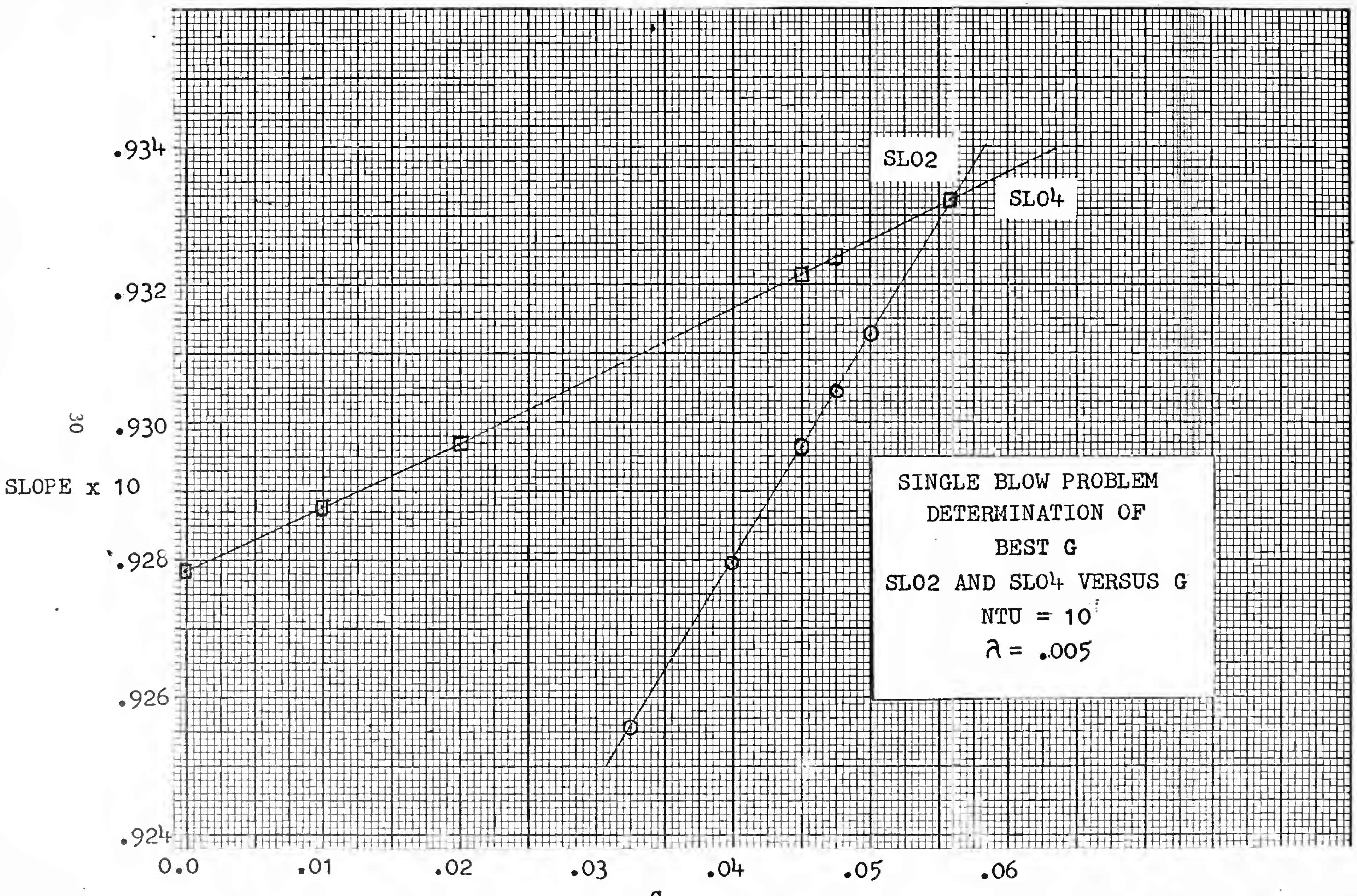

SL02
SL04
SINGLE BLOW PROBLEM
DETERMINATION OF
BEST G
SL02 AND SL04 VERSUS G
NTU = 10
λ = .005
SLOPE x 10
.934
.932
.930
.928
.926
.924
0.0
.01
.02
.03
.04
.05
.06
G

FIGURE 7

6. Discussion of Results.

Table 1 indicates that programs SL03 and SL05 are not sensitive to G and that both programs give the same solution to six decimal places for NTU greater than 5. The variation of G from -0.1 to +0.1 is much larger than the variations used in the SL02 and SL04 programs which are more sensitive to G. All results presented were calculated with eight point formulas.

One way to increase the accuracy of the program is to increase the number of terms in the partial sum. There is a test in the program which will increment the number of partial sums if the nth average has not converged to within 1×10^{-9} of the preceeding nth average for each calculation of slope. The test for the sufficient number of partial sums indicates that for NTU greater than five, twenty partial sums will give the same accuracy as 100 partial sums for programs SL03 and SL05. At NTU of five, the SL03 program automatically increased the number of partial sums to 25 which then gave as accurate a solution as 95 partial sums. There is an indication that for lower values of NTU, more partial sums may be necessary.

R. E. Maxim [10] presented a table of values of maximum slopes calculated with Schumann's solution for the special case of $\lambda = 0.0$. Schumann's solution is an infinite series of Bessel functions. Each value of slope was calculated using Bessel functions accurate to ten significant places and the summation was continued until there was no change in the tenth decimal place of the solution. Unfortunately, the search by Maxim could not distinguish between relative maximum slope values and his search ended when the slope at the next increment of time

was less than the preceeding slope. Relative maximum slopes have been found by the theoretical solution presented in this paper and their existence is supported by experimental evidence found by J. M. Bannon [1], Figure 26. The results of Table 2 clearly indicate that Maxim's values, though accurate to six places, were calculated at a time much too low to have found the largest relative maximum slope in the NTU range less than five. For values of NTU equal to and greater than five, both the time at which maximum slope occurred and the values of maximum slope times NTU calculated by SLO3 agree to at least seven decimal places with Maxim's results. SLO5 solutions were in complete agreement with the results of SLO3.

The computation time depends on the degree of accuracy required to calculate one value of slope. The accuracy required of the search routine for the results of Table 2 was $\pm 1 \times 10^{-9}$ difference in the new value of maximum slope as compared to the previous iteration. For values of NTU less than ten, the number of iterations required was four or five times the number for NTU values greater than ten. Once the approximate value of time at maximum slope is known, the search routine can be set to find the maximum with fewer iterations. But, even without optimum starting times the computation time was approximately half a minute for the $\lambda = 0.0$ case. For non zero values of λ, the computation time per search is expected to be twelve minutes.

Results of SLO5 for $\lambda = .04$ are shown to be insensitive to G in Table 3. One test point at NTU = 10 indicates good agreement between SLO5 and SLO3 programs. As for the $\lambda = 0$ case, results indicate good agreement with C. P. Howard's results [5]. For NTU greater than five Howard's results are consistantly low but all have less than 1% difference as compared to SLO5. As in the $\lambda = 0$ case, Howard's results for

NTU equal to two are much lower than those obtained by SL05. Again this is probably a failure of his search technique to distinguish between relative maximum slopes. Unfortunately, he did not publish the times at which the maximum slopes occurred. Howard also used the incremental step search and the search ended when the newly calculated slope was less than the preceeding slope.

Programs SL02 and SL04 are quite sensitive to parameter G. The results obtained graphically and listed in Table 4 were compared to the accurate solution of Maxim for λ = 0.0. The percent error of the graphical results was less than $\pm$ 1.5% except at NTU = 2. Figure 6 indicates that this intercept was not determined. Considerable effort would be required to find this intercept and obviously the mean value at NTU = 2 has unacceptably large error. For the other values of NTU, however, intercepts were easily found as indicated by Figure 5.

Table 5 shows similar results for λ = 0.005. No accurately known values are available for this region. C. P. Howard's results are compared to the results obtained by SL02 and SL04. If as expected, the SL02 and SL04 results are 1 to 1.5% too large, then C. P. Howard's solutions for λ = 0.005 would be approximately .5% too low for NTU greater than 10. Figure 7 again shows how the intercept or best G was obtained.

7. Conclusions and Recommendations.

Program SL05, using the VR, Gauss-Legendre formulas, is a fast accurate tool capable of supplying all the theoretical maximum slope data needed for experimental testing of compact heat exchanger surfaces. The SL03, VI, program is equally accurate but slightly slower. The SL05 program is being applied to verify the NTU versus Maximum Slope curves presented by C. P. Howard [5] . The results of these calculations will be published at a later date.

The combination of Laplace transforms with numerical inversion should be applicable to most boundary value problems in which one is able to express the solution as an inverse Laplace transform. Therefore, this technique is primarily limited in application to linear systems of partial differential equations with constant coefficients.

The two main problems encountered in the numerical inversion were the slow convergence of the integration and radical behavior of the function of s at time equal zero. The VR and VI functions were cyclic in nature with slowly decreasing magnitudes and period. For this particular solution the Legendre polynomial approximated the functions better than Chebyshev's polynomial. However, for other functions a number of other polynomials may give better results. The acceleration of convergence is adequately accomplished by the present numerical inversion but it generates a problem in addition. Because of the averaging technique applied to the partial sums to accelerate convergence the formal error analysis normally applied to each partial sum is impossible. Therefore, the only way remaining to determine the accuracy of the calculated maximum slopes was by comparison with R. E. Maxim's accurately determined values for the

limiting case of $\mathcal{R} = 0$. Even though direct inversion of the Laplace transform was possible for this case, the numerical inversion was accomplished in seconds with the same accuracy attained by the approximation to an infinite sum of Bessel functions carried out to six decimal place accuracy. For non-zero $\mathcal{R}$, there were no accurate solutions with which to compare. For these values an indication of accuracy was obtained by calculating the same points with an independent program. The SL03 program is not independent of SL05 in a strict interpretation of independence. However, they are different programs, SL03 using the imaginary part of the inversion integral and SL05 using the real part. Either program should give the same result. The same result to six or seven decimal places was obtained with these programs indicating excellent accuracy for a numerical process. The comparison of SL05 results with Howard's results, obtained with a rather crude finite-difference technique, indicated that he obtained very accurate results by judicious application of the finite-difference technique. Most of his results were less than one half of one percent low when compared to SL05.

The following recommendations indicate avenues of investigation that could be undertaken with either computer programs developed in conjunction with this thesis or the mathematical technique presented.

It is recommended that the temperature programs TEMP4 and TEMP2 be applied to obtain a time-temperature history for comparison with corresponding experimental results to determine how well the mathematical model represents the behavior of the experimental test rig.

An investigation should be conducted to determine the cause of and physical significance of the relative maximum slopes obtained both experimentally and theoretically for the heating cycle. One objective would be

to determine if the relative maximum obtained experimentally correspond in magnitude and time of occurance with those obtained by theory. Experimentally it was noted that the relative maximums are practically indistinguishable for the cooling of the solid, but that for the same matrix and mass flow of fluid the relative maximums are very distinct for the case of heating the solid. An investigation should be made to determine theoretically the effect of longitudinal conduction for the cooling cycle as compared to the present solution for the heating cycle. The objective of this investigation would be to determine if the theoretical solution is capable of predicting a difference in response on cooling as compared to heating. The cooling cycle solution would require appropriate sign changes in the heat balance from which governing partial differential equations were derived. This may, therefore, require a new solution for cooling since the sign changes could cause a change in the auxiliary cubic equation 3b of Section 3. It is possible that a sign change of a term in this equation could cause a change in the significance of the parameter λ on the result of fluid temperature or slope of fluid temperature. Bannon [1] has noted experimentally that the largest maximum slope obtained by cooling agreed fairly well with that obtained by heating.

The theoretical results, in the region of NTU = 2, are difficult to obtain, and the determination of NTU given an experimental maximum slope in this NTU region is impossible. An attempt should be made to develop a curve matching technique employing a least square error method. This method would use three or more values of time and their corresponding temperatures from an experimental time-temperature history. The TEMP4 program would calculate theoretical values of temperature at the given

times for an estimated value of NTU. Then an NTU search routine could calculate the sum of the squared error of each temperature. After repeating this operation for two bracketing values of NTU, a Lagrange iteration method would calculate the NTU with the least squared error. Another possible means of curve matching would be to match the dimensionless time at which the maximum slope occurs for the region of NTU = 2, rather than matching the magnitudes of maximum slope. This would require an adequate starting mark on the experimental time-temperature history and a correction of the time at the maximum slope to account for the system time delay of the experimental response.

A more general system of governing partial differential equations can be obtained from the heat balance as derived by Creswick, by including a term to represent the energy stored in the fluid. This term is required for transient heat transfer testing using a liquid rather than a gas. The result of this addition changes equation 2 of section 3 to the following

$$\frac{\partial v}{\partial X} = \frac{1}{NTU}\left[u - v - \lambda_f \frac{\partial v}{\partial t}\right],$$

where $\lambda_f \triangleq \dfrac{\gamma C_f A_f \ell}{W_s C_s}$, the dimensionless fluid capacitance, and γ is the density of the fluid in $lb/ft.^3$ This equation combined with Equation 1 of Section 3 would then be the new governing partial differential equations. The present mathematical technique should be capable of solving this system of equations.

BIBLIOGRAPHY

1. Bannon, J. M., "An Experimental Determination of Heat Transfer and Flow Characteristics of Perforated Material for Compact Heat Exchanger Surfaces," Master Thesis, 1964, U. S. Naval Postgraduate School.

2. Churchill, R. V., Operational Mathematics, McGraw-Hill, 1958, second edition.

3. Creswick, F. A., "A Digital Computer Solution of the Equations for Transient Heating of a Porous Solid Including the Effects of Longitudinal Conduction," Industrial Mathematics, 1957, pp. 61-69.

4. Franklin, P. Methods of Advanced Calculus, McGraw-Hill, 1944.

5. Howard, C. P., "Heat Transfer and Flow Friction Characteristics of Skewed Passage and Glass-Ceramic Heat Transfer Surfaces," Department of Mechanical Engineering, Stanford University, TR-No. 59, Appendix 3, pp. 31-43., Oct., 1963.

6. Hurwitz, H., Jr., and P. F. Sweifel, "Numerical Quadrature of Fourier Transform Integrals," Math Tables Aids Comp., X, 1956, pp. 140-149.

7. Jakob, M., Heat Transfer, Vol. II, John Wiley and Sons, Inc., 1957.

8. Lanczos, C., Applied Analysis, Prentice Hall, Inc., 1956.

9. Locke, G. L., "Heat Transfer and Flow-Friction Characteristics of Porous Solids," Department of Mechanical Engineering, Stanford University, TR-No. 10, June 1950.

10. Maxim, R. E., "Convective Heat Transfer and Flow Friction Characteristics of Compact Heat Exchanger Surfaces," Master Thesis, M386, 1962, U. S. Naval Postgraduate School.

11. Mondt, J. R., "Effects of Longitudinal Thermal Conduction in the Solid on Apparent Convection Behavior, with Data for Plate-Fin Surfaces," International Heat Transfer Conference, Boulder, Colorado, Paper No. 73, Proceedings, ASME, 1961.

12. Romie, F. E., et al., "Heat Transfer and Pressure Drop Characteristics of Four Regenerative Heat Exchanger Matrices," Department of Engineering, University of California, December 1948, Contract No. a(S)-8649 for Bureau of Aeronautics.

13. Schmittroth, L. A., "Numerical Inversion of Laplace Transforms," Communications of the ACM, pp. 171-172.

14. Schumann, T. E. W., "Heat Transfer: A Liquid Flowing Through a Porous Prism," Journal of Franklin Institute, CCVIII, July-December, 1929, pp. 405-416.

APPENDIX I

Solution for Zero and Infinite Longitudinal Conduction.

1. ZERO LONGITUDINAL CONDUCTIVITY

As in the solution for finite conductivity the development of this case is the same initally; cf, Equation 3a, page 8.

(1) $$\frac{\partial^3 v(x,s)}{\partial x^3} + b\frac{\partial^2 v(x,s)}{\partial x^2} - \frac{b(s+1)}{a}\frac{\partial v(x,s)}{\partial x} - \frac{b^2}{a} s\, v(x,s) = 0$$

For this case where λ is defined as a, it is necessary to multiply by a. The substitution of a = 0 results in the equation

(2) $$v' + b\frac{s}{s+1} v = 0.$$

The general solution of equation 2 is

(3) $$v(x,s) = C_1(s)\, e^{r_1 x} = C_1(s)\, e^{-\frac{bs}{s+1}x}.$$

The boundary conditions are the same as in the previous solution. Boundary conditions a and b were applied in order to arrive at equation 1, above. Boundary condition C is v(0,t) = C which transforms into v(0,s) = C/s. Applying boundary condition c to equation 3 results in

$$v(0,s) = \frac{C}{s} = C_1(s),$$ therefore

(4) $$v(x,s) = \frac{C}{s} e^{-\frac{bs}{s+1}x}$$ and $$v(1,s) = \frac{C}{s} e^{-\frac{bs}{s+1}}$$

It follows directly,

(5) $$\frac{\partial v(x,s)}{\partial t} = Ce^{-b\frac{s}{s+1}x} \quad \text{and} \quad \frac{\partial v(1,s)}{\partial t} = Ce^{-\frac{bs}{s+1}}.$$

This special case was solved analytically in 1929 by T. E. W. Schumann [14] . The following development will show that the present solution agrees with Schumann's.

Let s' = s + 1 and apply it to equation 4.

Then
$$v(x, s'-1) = \frac{C}{s'-1} e^{-b\frac{s'-1}{s'}x}$$
or

(6) $$v(x,s'-1) = Ce^{-bx}\sum_{m=1}^{\infty} e^{\frac{bx}{s'}} \frac{1}{s'^m}$$

From page 229 of reference [2] , we find

(7) $$\mathcal{L}^{-1}\left\{\frac{1}{s^{\mu}} e^{\frac{K}{s}}\right\} = \left(\frac{t}{K}\right)^{\frac{\mu-1}{2}} I_{\mu-1}(2\sqrt{Kt}).$$

Comparing equation 6 and 7, the inverse transform of equation 6 becomes:

(8) $$\mathcal{L}^{-1}\{v(x,s')\} = Ce^{-bx}\sum_{m=1}^{\infty}\left(\frac{t}{bx}\right)^{\frac{m-1}{2}} I_{m-1}(2\sqrt{btx}).$$

From page 294 [2] is

$$\mathcal{L}^{-1}\{f(s-a)\} = e^{at}\cdot\mathcal{L}^{-1}\{f(s)\}$$
therefore,

(9) $$v(x,t) = Ce^{t}e^{-bx}\sum_{m=1}^{\infty}\left(\frac{t}{bx}\right)^{\frac{m-1}{2}} I_{m-1}(2\sqrt{btx}),$$

(9a) $$v(x,t) = Ce^{-b(x-\frac{t}{b})}\sum_{m=0}^{\infty}\left(\frac{t}{bx}\right)^{\frac{m}{2}} I_{m}(2\sqrt{btx}).$$

Define a = btx and W = 2 $\sqrt{a}$, then compare equation 9a with the form

$$\text{(9b)} \qquad a^{-\frac{m}{2}} I_m(2\sqrt{a}).$$

Note that

$$\text{(9c)} \quad \left(\frac{t}{bx}\right)^{\frac{m}{2}} = \frac{t^m}{(bxt)^{m/2}} = t^m a^{-\frac{m}{2}}$$

Equation 9a then becomes

$$\text{(10)} \quad v(x,t) = Ce^{-bx+t} \sum_{m=0}^{\infty} t^m a^{-\frac{m}{2}} I_m(2\sqrt{a}).$$

Schumann's solution is

$$\text{(11)} \quad v(y,z) = e^{-y-z} \sum_{m=0}^{\infty} z^m \frac{d^m J_0(2i\sqrt{yz})}{d(yz)^m} .$$

Equations 10 and 11 are equivalent which can be shown as follows. From page 392 [4] , the modified Bessel function identify is

$$\frac{d}{dW}\left[W^{-m} I_m(W)\right] = W^{-m} I_{m+1}(W) \qquad \text{where } m \geq 0$$

Then, for n = 0 and W = 2 $\sqrt{a}$,

$$\frac{d}{dW}\left[I_0(W)\right] = I_1(W)$$

Apply the chain rule such that

$$\frac{d[Y]}{da} = \frac{dY}{dW} \cdot \frac{dW}{da}$$

It follows that

$$\frac{d[I_0(2\sqrt{a})]}{da} = a^{-1/2} I_1(2\sqrt{a}).$$

By mathematical induction it can be shown that

(12) $$\frac{d^m}{da^m}\left[I_0(2\sqrt{a})\right] = a^{-\frac{m}{2}} I_m(2\sqrt{a}).$$

Equation 10 then becomes

(13) $$v(x,t) = Ce^{-bx+t} \sum_{m=0}^{\infty} t^m \frac{d^m}{da^m}\left[I_0(2\sqrt{a})\right]$$

From page 392 [4],

$$I_m(2\sqrt{a}) = i^{-m} J_m(2i\sqrt{a}),$$ and

$$I_0(2\sqrt{a}) = J_0(2i\sqrt{a}).$$

Note that Schumann's development is for the case of heating the fluid as time increases and that the present case is developed for cooling of the fluid with increasing time, which changes the sign of t in the exponential term   It then follows that equation 13 is, indeed,

(11) $$v(y,z) = e^{-y-z} \sum_{m=0}^{\infty} z^m \frac{d^m}{d(yz)^m}\left[J_0(2i\sqrt{yz})\right].$$

Since the program for numerical inversion of equations 5 and 6 is already available it is much easier and probably much faster to apply it than to evaluate equation 11. The same theoretical solution was obtained by G. L. Locke in 1950 [9].

2. INFINITE LONGITUDINAL CONDUCTIVITY

When $\lambda = \infty$, the solid temperature is no longer a function of x. In fact the following is true: $\frac{\partial^m u}{\partial x^m}(x,t) = 0.$ Therefore, equation 1, page 7, becomes indeterminate and must be replaced by:[7]

$$(1) \quad \frac{du(t)}{dt} = \int_0^1 \left[v(x,t) - u(t)\right] dx \qquad \text{and equation 2 is}$$

$$(2) \quad \frac{\partial v(x,t)}{\partial x} = NTU\left[u(t) - v(x,t)\right].$$

The boundary conditions to be applied to this case are:

a. $v(x,0) = 0$,

b. $v(x,0) = 0$, and

c. $v(0,s) = C/s$.

Solve equation 2 for u(t) and take the derivative of u with respect to t. Then substituting these back into equation 1 and applying boundary condition c, results in

$$\frac{1}{NTU}\frac{\partial^2 v(x,t)}{\partial x \partial t} + \frac{\partial v(x,t)}{\partial t} = \int_0^1 \left[v(x,t) - \frac{1}{NTU}\frac{\partial v}{\partial x}(x,t) - v(x,t)\right] dx$$

$$= \frac{-1}{NTU}\left[v(1,t) - C\right].$$

Multiply through by NTU and rearrange the terms to arrive at

$$(3) \quad \frac{\partial^2 v(x,t)}{\partial x \partial t} + NTU\frac{\partial v(x,t)}{\partial t} + v(1,t) - C = 0.$$

Transformation of each term in 3 by Laplace gives

[7]For further discussion, see page 47.

$$(4)\quad -\frac{\partial v(x,0)}{\partial x} + \frac{s\,\partial v(x,s)}{\partial x} + NTU\left[-v(x,0) + s\,v(x,s)\right] + v(1,s) - \frac{C}{s} = 0$$

Applying boundary conditions a and b and dividing by s, equation 4 reduces to

$$(5)\quad \frac{\partial v(x,s)}{\partial x} + NTU\, v(x,s) = \frac{C}{s^2} - \frac{v(1,s)}{s}\,.$$

Again by treating s as a parameter, equation 5 is a total differential equation. The solution of the differential equation is

$$(6)\quad v(x,s) = C_1(s)\, e^{-(NTU)x} + \frac{1}{NTU}\cdot\frac{1}{s}\left[\frac{C}{s} - v(1,s)\right].$$

Now apply boundary condition c to equation 6 which results in

$$(7)\quad v(0,s) = \frac{C}{s} = C_1(s) + \frac{1}{NTU}\cdot\frac{1}{s}\left[\frac{C}{s} - v(1,s)\right]$$

Therefore, $C_1(s) = \frac{C}{s}\left[1 - \frac{1}{NTU\cdot s}\right] + \frac{1}{NTU\cdot s}\, v(1,s)$, and

$$(8)\quad v(x,s) = \frac{C}{s}\, e^{-NTU\cdot x} + \frac{1}{NTU\cdot s}\left[\frac{C}{s} - v(1,s)\right]\left(1 - e^{-NTU\cdot x}\right)$$

Solve equation 8 for v(1,s) as follows:

$$v(1,s) = \frac{C}{s}\, e^{-NTU} + \frac{1}{NTU\cdot s}\left[\frac{C}{s} - v(1,s)\right]\left(1 - e^{-NTU}\right)$$

$$v(1,s)\left[1 + \frac{(1 - e^{-NTU})}{NTU\cdot s}\right] = \frac{C}{s}\left[\frac{(1 - e^{-NTU})}{NTU\cdot s} + e^{-NTU}\right].$$

Define $\quad \xi = \dfrac{(1-e^{-NTU})}{NTU}$; then $e^{-NTU} = 1 - \xi\, NTU$, and

(9) $$\upsilon(1,s) = \frac{C}{s}\;\frac{\left(1 + \frac{\xi}{s} - \xi \cdot NTU\right)}{(1 + \xi/s)}$$

The particular fluid temperature of interest is that temperature at x = 1. Also the value of C must be dimensionless, for comparison with Mondt's solution, and can be set equal to one. It follows directly that

$$\upsilon(1,s) = 1\,\frac{\left(1 + \frac{\xi}{s} - \xi \cdot NTU\right)}{(s + \xi)}$$

(10) $$\upsilon(1,s) = \frac{1}{s+\xi} - \frac{\xi \cdot NTU}{s+\xi} + \frac{\xi}{s(s+\xi)} .$$

The inverse Laplace transform of equation 10 is

$$\upsilon(1,t) = e^{-\xi t} - NTU \cdot \xi e^{-\xi t} + 1 - e^{-\xi t}$$ which reduces to

(11) $$v(1,t) = 1 - NTU\,\xi\, e^{-\xi t},$$

where $\xi = \dfrac{1-e^{-NTU}}{NTU}$. The solution as a function of time is

$$v(1,t) = 1 - (1-e^{-NTU})\, e^{-\frac{(1-e^{-NTU})}{NTU} t}.$$

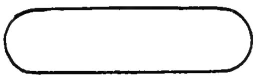
This solution is in complete agreement with that derived by J. R. Mondt [11] in 1961. Since a convenient inverse transform was available it was unnecessary to apply the intricate numerical inversion.

To derive Equation (1) for this special case it was necessary to refer to the heat balance from which Creswick [3] derived the governing equations. The last term of Creswick's Equation (1) was eliminated and the remaining terms were arranged in terms of dimensionless parameters. Then both sides of the equation were integrated over the length of the matrix to arrive at Equation (1) on page 44.

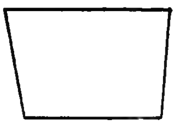
APPENDIX II

Flow diagrams and program listings.

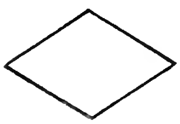
The flow diagrams will preceed the applicable program listings. The symbols used in the flow diagrams are defined as follows:



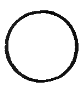
Terminal - the beginning, end, or point of interuption in the program



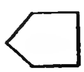
Input/Output - input symbols are preceeded by a terminal.



Decision - branches are labeled by sign according to the sign of the value enclosed.

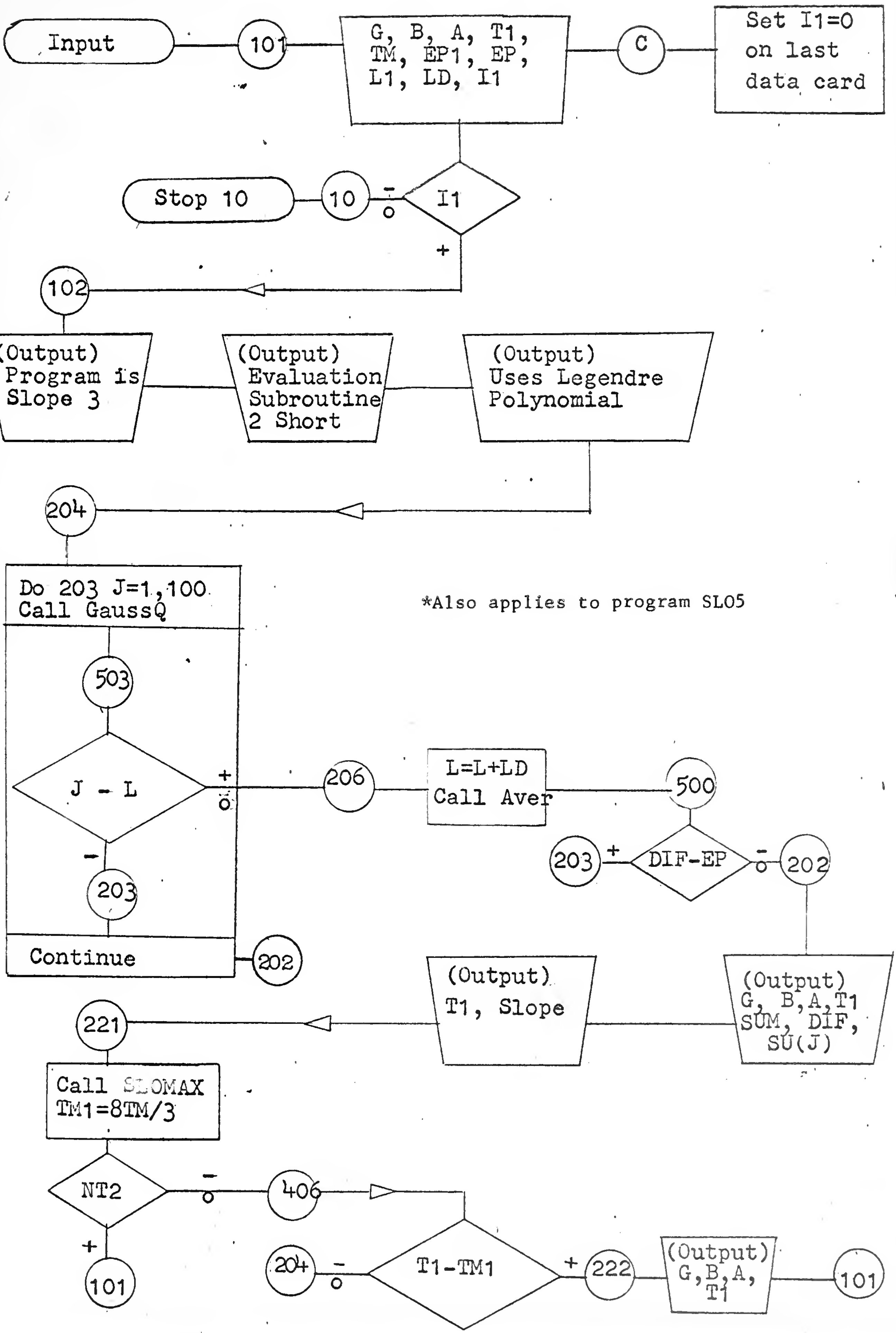


Connector - enclosed numbers refer to the prefix numbers of the corresponding statements on the program listing.



Offpage Connector - designates an entry to or exit from a page.

There are six distinct programs which are used separately, although they could be combined into a larger package to make use of one or another at the user's option. There are the four "slope" programs, SLO2, SLO3, SLO4 and SLO5 which have been described in detail in the body of the thesis, and the two "temperature" programs TEMP2 and TEMP4, which have been referred to in the text but not described in detail. The latter are intended to provide a time-temperature history; no use of these programs has been made in this thesis. See page 35 for recommendations concerning their use.

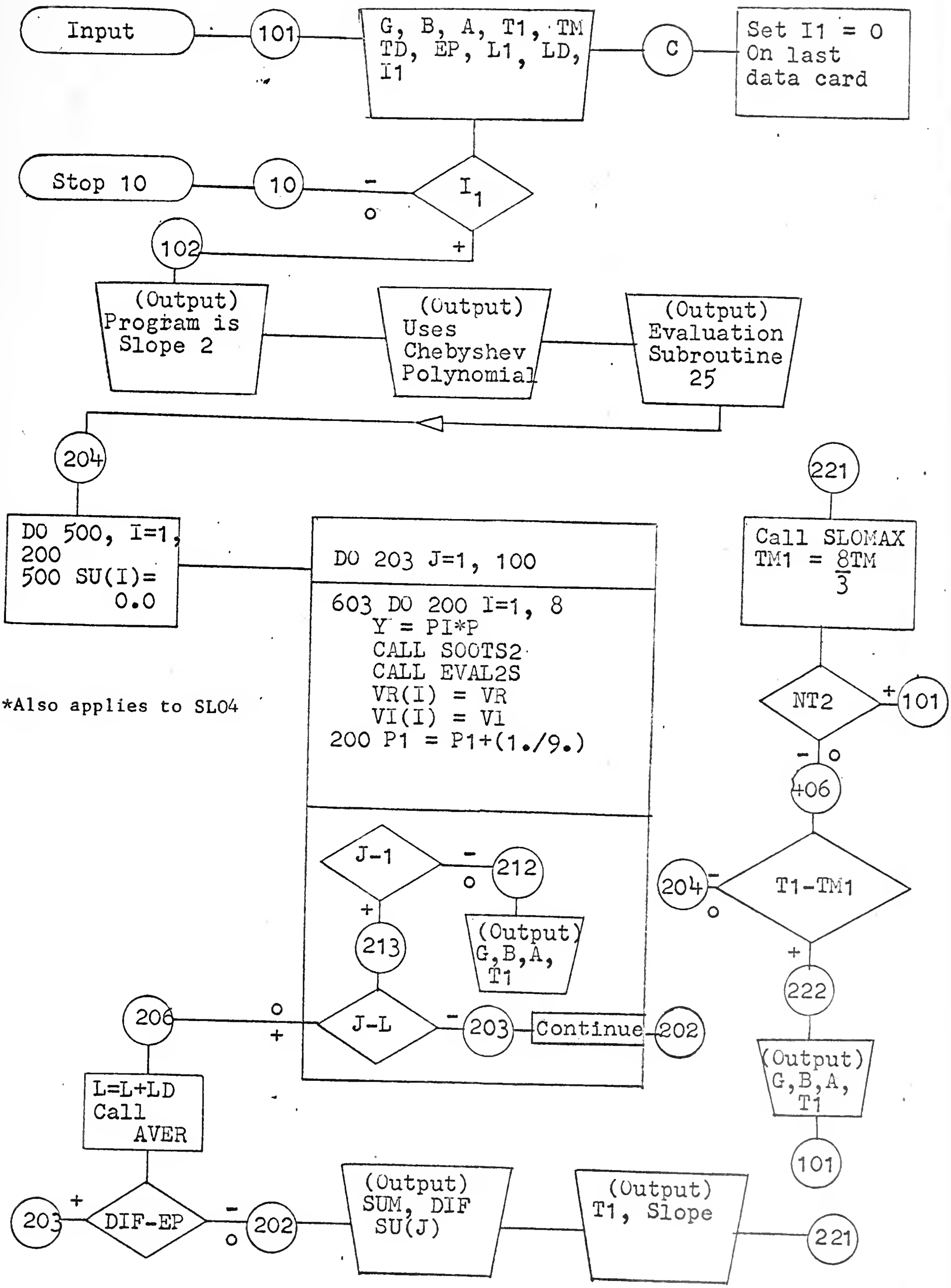


*Also applies to program SL05

```
      PROGRAM SLO3
      DIMENSION VR(20),VI(20),SU(200),AV(12,10),R(129),X(129),ER(200),
     1RR(128),RI(128),T(10),SLO(10)
      COMMON RR,RI,T,SLO,G,B,A,PI,SLOPE,NT,NT1,NT2,EP1,I1,T1
      COMMON NT3,TM
  101 READ 100,G,B,A,T1,TM,EP1,EP,L1,LD,I1
  100 FORMAT(7F10.5,3I3)
C     SET I1 = 0  ON LAST DATA CARD
C     I1 IS THE AVERAGE AND THE POLYNOMIAL ORDER
      IF(I1)10,10,102
  102 CONTINUE
      NT=-1
      NT2=0
      K=1
      NT1=-1
      PI=3.1415926536
      WRITE OUTPUT TAPE 4,301
  301 FORMAT(24H PROGRAM IS SLOPE 1,' N=8//)
      WRITE OUTPUT TAPE 4,302
  302 FORMAT(30H EVALUATION SUBROUTINE 2 SHORT/)
      WRITE OUTPUT TAPE 4,305
  305 FORMAT(25H USES LEGENDRE POLYNOMIAL/)
  204 L=L1
      NT3=-1
      SU=0.
      D=0.0
      E=1.0
      DO 203 J=1,100
      CALL GAUSSQ(D,E,VAL)
      D=D+1.0
      E=E+1.0
      SU(J)=VAL+SU(J-1)
  503 IF(J-L)203,206,206
  206 L=L+LD
      CALL AVER(SU,AV,I1,J,DIF,SUM)
  500 IF(DIF-EP)202,202,203
  203 CONTINUE
      CALL AVER(SU,AV, 8,100,DIF,SUM)
  202 EG=EXPF(G*T1)
      EG2=2.*EG/T1
      SLOPE=EG2*SUM
      PRINT 3,G,B,A,T1
      PRINT 4,SUM,DIF,J,SU(J)
      PRINT 2,T1,SLOPE
  221 CALL SLOMAX
      TM1=8.*TM/3.
      IF(NT2)406,406,101
  406 IF(T1-TM1)204,204,222
  222 PRINT 3,G,B,A,T1
    2 FORMAT(3H T=F10.5,3X,6HSLOPE=E20.8/)
    3 FORMAT(3H G=F10.5,3X,2HB=F10.5,3X,2HA=F10.5,3X,2HT=F10.5/)
    4 FORMAT(5H SUM=E20.8,3X,4HDIF=F10.9,3X,3HSU(I3,2H)=E20.8/)
      GO TO 101
   10 STOP 10
      END
```

```
      PROGRAM SL05
      DIMENSION VR(20),VI(20),SU(200),AV(12,10),R(129),X(129),ER(200),
     1RR(128),RI(128),T(10),SLO(10)
      COMMON RR,RI,T,SLO,G,B,A,PI,SLOPE,NT,NT1,NT2,EP1,I1,T1
      COMMON NT3,TM
  101 READ 100,G,B,A,T1,TM,EP1,EP,L1,LD,I1
  100 FORMAT(7F10.5,3I3)
      SET I1 = 0  ON LAST DATA CARD
      I1 IS THE AVERAGE AND THE POLYNOMIAL ORDER
      IF(I1)10,10,102
  102 CONTINUE
      NT=-1
      NT2=0
      K=1
      NT1=-1
      PI=3.1415926536
      WRITE OUTPUT TAPE 4,301
  301 FORMAT(24H PROGRAM IS SLOPE 5, N=8/)
      WRITE OUTPUT TAPE 4,302
  302 FORMAT(30H EVALUATION SUBROUTINE 2 SHORT/)
      WRITE OUTPUT TAPE 4,305
  305 FORMAT(25H USES LEGENDRE POLYNOMIAL/)
  204 L=L1
      NT3=-1
      SU=0.
      D=0.0
      E=1.0
      DO 203 J=1,100
      CALL GAUSSQ(D,E,VAL)
      D=D+1.0
      E=E+1.0
      SU(J)=VAL+SU(J-1)
  503 IF(J-L)203,206,206
  206 L=L+LD
      CALL AVER(SU,AV,I1,J,DIF,SUM)
  500 IF(DIF-EP)202,202,203
  203 CONTINUE
      CALL AVER(SU,AV, 8,100,DIF,SUM)
  202 EG=EXPF(G*T1)
      EG2=2.*EG/T1
      SLOPE=EG2*SUM
      PRINT 3,G,B,A,T1
      PRINT 4,SUM,DIF,J,SU(J)
      PRINT 2,T1,SLOPE
  221 CALL SLOMAX
      TM1=8.*TM/3.
      IF(NT2)406,406,101
  406 IF(T1-TM1)204,204,222
  222 PRINT 3,G,B,A,T1
    2 FORMAT(3H T=F10.5,3X,6HSLOPE=E20.8/)
    3 FORMAT(3H G=F10.5,3X,2HB=F10.5,3X,2HA=F10.5,3X,2HT=F10.5/)
    4 FORMAT(5H SUM=E20.8,3X,4HDIF=F10.9,3X,3HSU(I3,2H)=E20.8/)
      GO TO 101
   10 STOP 10
      END
```

PROGRAM SL02*

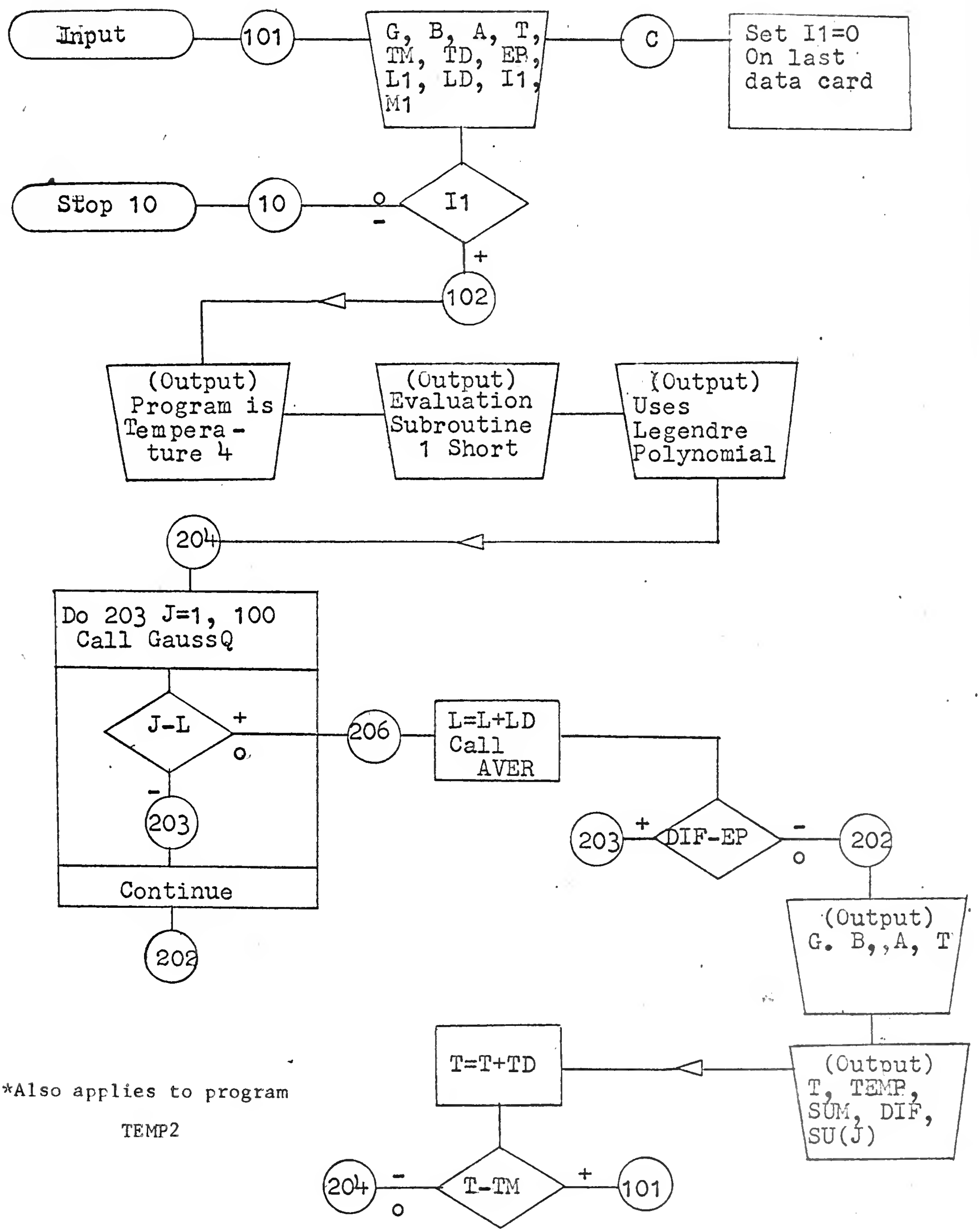


*Also applies to SL04

```
      PROGRAM SLO2
      DIMENSION RR(128),RI(128),VI(20),VR(20),SU(200),AV(12,10),R(129),
     1X(129),SLO(10),T(10)
      COMMON SLOPE,SLO,T,NT,NT1,NT2,K,EP1,G,B,A,PI,T1,TM
      COMMON NT3
  101 READ 100,G,B,A,T1,TM,TD,EP,L1,LD,I1
  100 FORMAT(7F10.5,3I3)
C     SET I1 = 0  ON LAST DATA CARD
      IF(I1)10,10,102
  102 C1=1.0
  210 WRITE OUTPUT TAPE 4,301
  301 FORMAT(24H PROGRAM IS SLOPE 2, N=4)
      WRITE OUTPUT TAPE 4,304
  304 FORMAT(26H USES CHEBYSHEV POLYNOMIAL)
  401 WRITE OUTPUT TAPE 4,302
  302 FORMAT(30H EVALUATION SUBROUTINE 2S      /)
      W1C=.10776070/.98480775
      W2C=.083333333/.86602540
      W3C=.045908434/.64278761
      W4C=.012997531/.34202014
      NT=-1
      NT1=-1
      NT2=0
      K=1
      EP1=.000001
      NT3=-1
  204 P=3.1415926536/T1
      DO 500 I=1,200
  500 SU(I)=0.0
      L=L1
      P1=1./9.
      SGN=-1.
      SU=0.
      DO 203 J=1,100
  603 DO 200 I=1,8
      Y=P1*P
      CALL SOOTS2 (G,B,A,Y,R,X,RR,RI)
      CALL EVAL2S(RR,RI,G,B,Y,C1,VR,VI)
      VR(I)=VR
      VI(I)=VI
  200 P1=P1+(1./9.)
  605 SU(J)=SGN*(W1C*(VI(4)+VI(5))+W2C*(VI(3)+VI(6))+W3C*(VI(2)+VI(7))+
     1W4C*(VI(1)+VI(8)))+SU(J-1)
      IF(J-1) 212,212,213
  212 PRINT 3,G,B,A,T1
  213 CONTINUE
      SGN=-SGN
      P1=P1+(2./9.)
      IF(J-L)203,206,206
  206 L=L+LD
      CALL AVER(SU,AV,I1,J,DIF,SUM)
      IF(DIF-EP)202,202,203
  203 CONTINUE
      CALL AVER(SU,AV,10,100,DIF,SUM)
  202 EG=EXPF(G*T1)
      EG2=2.*EG/T1
      SLOPE=EG2*SUM
      PRINT 4,SUM,DIF,J,SU(J)
      PRINT 2,T1,SLOPE
  221 CALL SLOMAX
      TM1=8.*TM/3.
      IF(NT2)406,406,101
  406 IF(T1-TM1)204,204,222
  222 PRINT 3,G,B,A,T1
    2 FORMAT(3H T=F10.5,3X,6HSLOPE=E15.8//)
    3 FORMAT(3H G=F10.5,3X,2HB=F10.5,3X,2HA=F10.5,3X,2HT=F10.5/)
    4 FORMAT(5H SUM=E15.8,3X,4HDIF=F10.9,3X,3HSU(I3,2H)=E15.8/)
      GO TO 101
   10 STOP 10
      END
```

```
      PROGRAM SLO4
      DIMENSION RR(128),RI(128),VI(20),VR(20),SU(200),AV(12,10),R(129),
     1X(129),SLO(10),T(10)
      COMMON SLOPE,SLO,T,NT,NT1,NT2,K,EP1,G,B,A,PI,T1,TM
      COMMON NT3
  101 READ 100,G,B,A,T1,TM,TD,EP,L1,LD,I1
  100 FORMAT(7F10.5,3I3)
C     SET I1 = 0  ON LAST DATA CARD
      IF(I1)10,10,102
  102 C1=1.0
  210 WRITE OUTPUT TAPE 4,301
  301 FORMAT(24H PROGRAM IS SLOPE 4, N=4)
      WRITE OUTPUT TAPE 4,304
  304 FORMAT(26H USES CHEBYSHEV POLYNOMIAL)
  401 WRITE OUTPUT TAPE 4,302
  302 FORMAT(30H EVALUATION SUBROUTINE 2S      /)
      W1C=.10776070/.98480775
      W2C=.083333333/.86602540
      W3C=.045908434/.64278761
      W4C=.012997531/.34202014
      NT=-1
      NT1=-1
      NT2=0
      K=1
      EP1=.000001
      NT3=-1
  204 P=3.1415926536/T1
      DO 500 I=1,200
  500 SU(I)=0.0
      L=L1
      P1=1./9.
      SGN=1.
      SU=0.
      DO 203 J=1,100
      IF(J-1)600,600,601
  600 P1=-7./18.
      GO TO 603
  601 IF(J-2)602,602,603
  602 P1=11./18.
  603 DO 200 I=1,8
      Y=P1*P
      CALL SOOTS2 (G,B,A,Y,R,X,RR,RI)
      CALL EVAL2S(RR,RI,G,B,Y,C1,VR,VI)
      VR(I)=VR
      VI(I)=VI
  200 P1=P1+(1./9.)
      IF(J-1)604,604,605
  604 SU(1)=.5*SGN*(W1C*(VR(4)+VR(5))+W2C*(VR(3)+VR(6))+W3C*( VR(2)+VR(7)
     1)+W4C*(VR(1)+VR(8)))+SU(J-1)
      GO TO 212
  605 SU(J)=SGN*(W1C*(VR(4)+VR(5))+W2C*(VR(3)+VR(6))+W3C*(VR( 2)+VR(7))+
     1W4C*(VR(1)+VR(8)))+SU(J-1)
      IF(J-1) 212,212,213
  212 PRINT 3,G,B,A,T1
  213 CONTINUE
      SGN=-SGN
      P1=P1+(2./9.)
      IF(J-L)203,206,206
  206 L=L+LD
      CALL AVER(SU,AV,I1,J,DIF,SUM)
      IF(DIF-EP)202,202,203
  203 CONTINUE
      CALL AVER(SU,AV,10,100,DIF,SUM)
  202 EG=EXPF(G*T1)
      EG2=2.*EG/T1
      SLOPE=EG2*SUM
      PRINT 4,SUM,DIF,J,SU(J)
      PRINT 2,T1,SLOPE
  221 CALL SLOMAX
      TM1=8.*TM/3.
      IF(NT2)406,406,101
  406 IF(T1-TM1)204,204,222
  222 PRINT 3,G,B,A,T1
    2 FORMAT(3H T=F10.5,3X,6HSLOPE=E15.8//)
    3 FORMAT(3H G=F10.5,3X,2HB=F10.5,3X,2HA=F10.5,3X,2HT=F10.5/)
    4 FORMAT(5H SUM=E15.8,3X,4HDIF=F10.9, 3X,3HSU(I3,2H)=E15.8/)
      GO TO 101
   10 STOP 10
      END
```

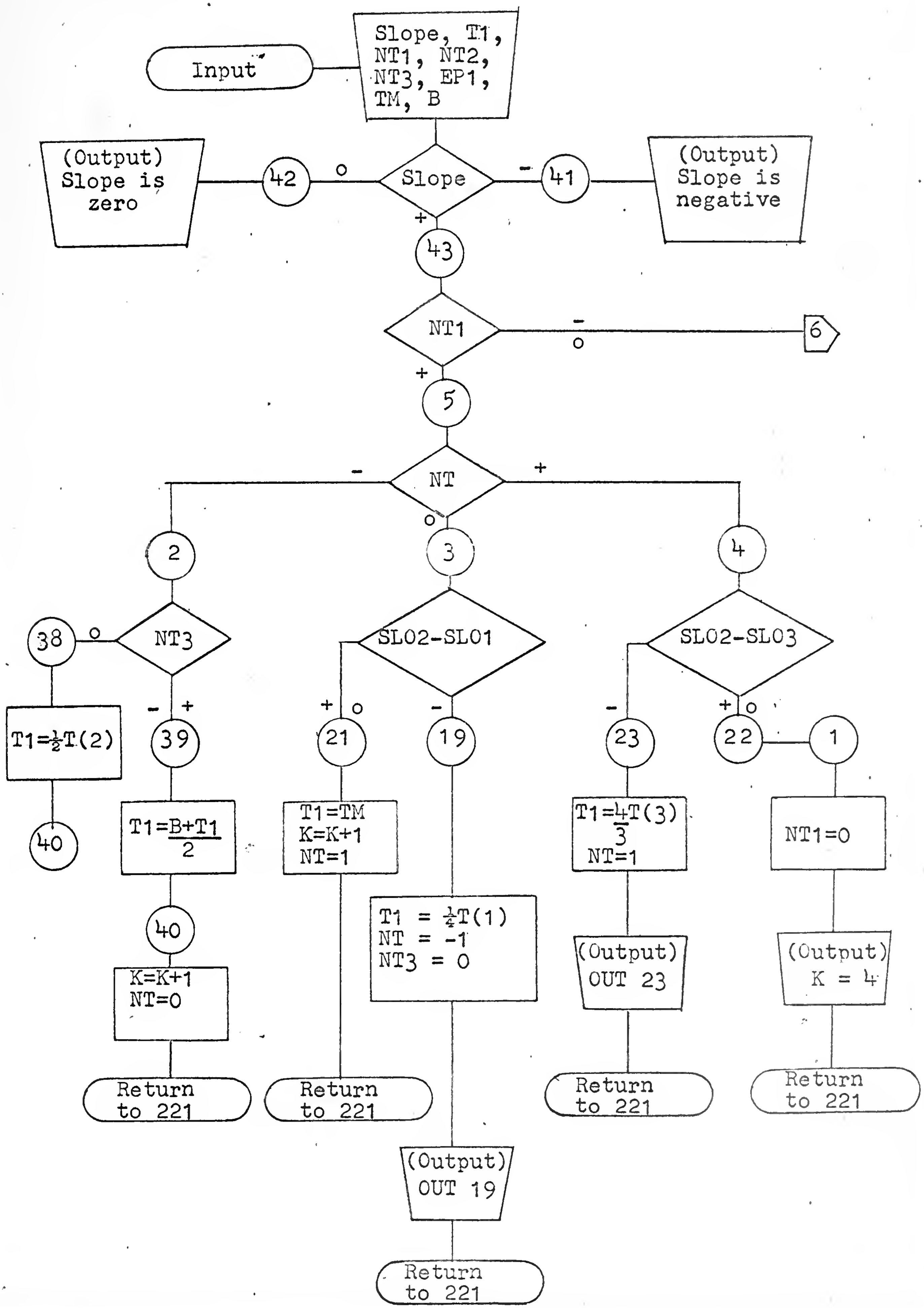
PROGRAM TEMP4*

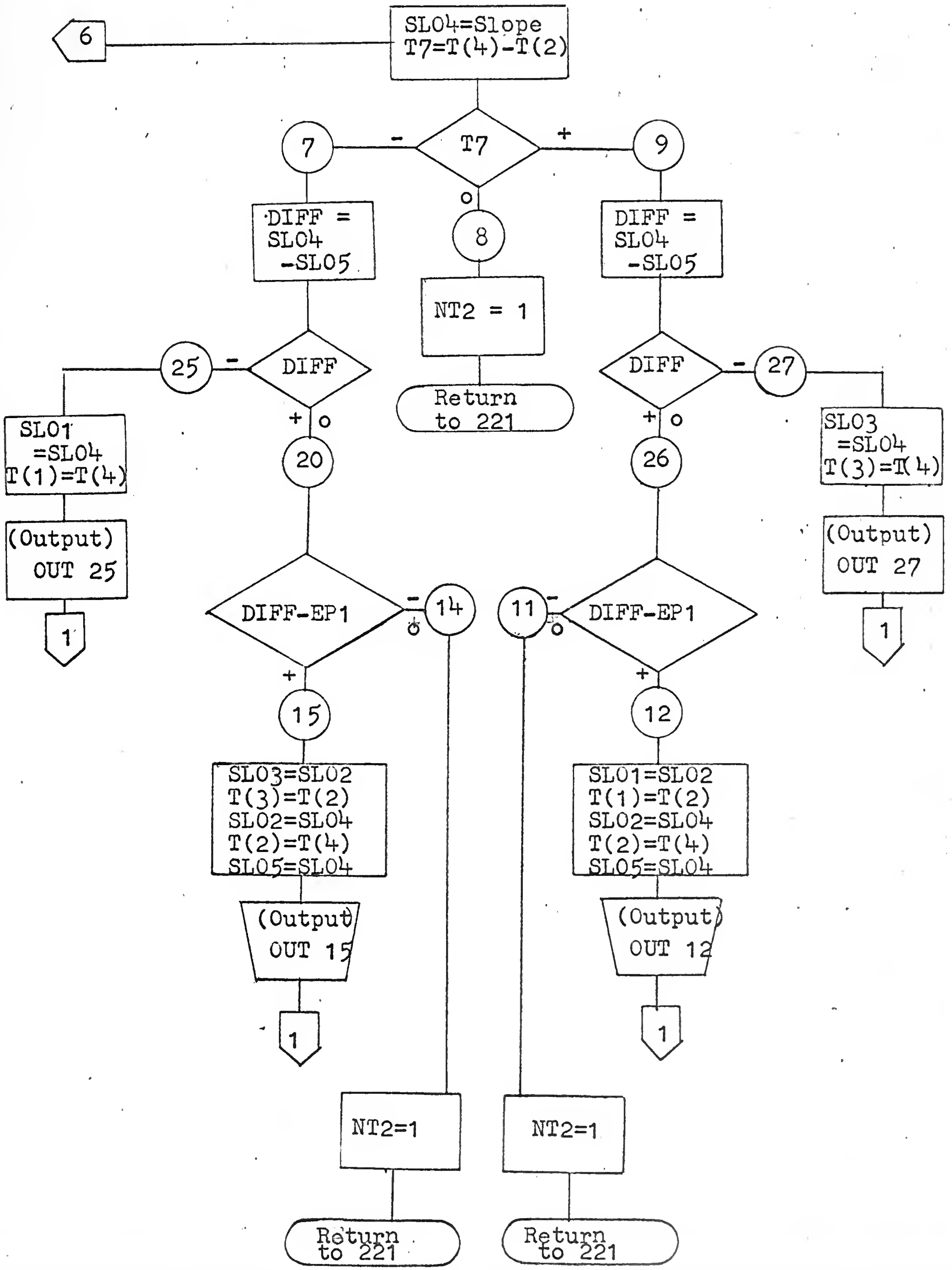


*Also applies to program TEMP2

```
      PROGRAM TEMP4*
      DIMENSION RR(128),RI(128),VI(20),VR(20),SU(200),AV(12,10),R(129),
     1X(129),ER(200)
      COMMON G,B,A,T,TM,TD,EP,L1,LD,I1,M1,RR,RI,PI
  101 READ 100,G,B,A,T,TM,TD,EP,L1,LD,I1,M1
  100 FORMAT(7F10.5,3I3,I1)
      SET I1 = 0  ON LAST DATA CARD
      IF(I1)10,10,102
  102 CONTINUE
      PI=3.1415926536
      WRITE OUTPUT TAPE 4,300
  300 FORMAT(30H PROGRAM IS TEMPERATURE 4, N=8/)
      WRITE OUTPUT TAPE 4,302
  302 FORMAT(30H EVALUATION SUBROUTINE 1 SHORT/)
      WRITE OUTPUT TAPE 4,305
  305 FORMAT(25H USES LEGENDRE POLYNOMIAL/)
  204 L=L1
      L=L1
      SU=0.
      D=0.0
      E=1.0
      DO 203 J=1,100
      CALL GAUSSQ(D,E,VAL,ER)
      D=D+1.0
      E=E+1.0
      SU(J)=VAL+SU(J-1)
      IF(J-L)203,206,206
  206 L=L+LD
      CALL AVER(SU,AV,I1,J,DIF,SUM)
      IF(DIF-EP)202,202,203
  203 CONTINUE
      CALL AVER(SU,AV, 8,100,DIF,SUM)
  202 EG=EXPF(G*T)
      EG2=2.*EG/T
      TEMP= EG2*SUM
      WRITE OUTPUT TAPE 4,1,G,B,A,T
    1 FORMAT(3H G=F10.5,3X,2HB=F10.5,3X,2HA=F10.5,3X,2HT=F10.5/)
      WRITE OUTPUT TAPE 4,2,T,TEMP,SUM,DIF,J,SU(J)
    2 FORMAT(3H T=F10.5,3X,5HTEMP=E20.8,3X,4HSUM=E20.8,3X,4HDIF=F10.9,
     13X,3HSU(I3,2H)=E20.8////)
      T=T+TD
      IF(T-TM)204,204,101
      GO TO 101
   10 STOP 10
      END
```

*TEMP2 is the same except statement 300.


Input
Slope, T1, NT1, NT2, NT3, EP1, TM, B
Slope
o
-
+
42
41
43
(Output) Slope is zero
(Output) Slope is negative
NT1
-
o
+
6
5
NT
-
+
o
2
3
4
NT3
o
-
+
38
39
T1=½T(2)
40
T1=(B+T1)/2
40
K=K+1
NT=0
Return to 221
SL02-SL01
+
o
-
21
19
T1=TM
K=K+1
NT=1
Return to 221
T1 = ¼T(1)
NT = -1
NT3 = 0
(Output) OUT 19
Return to 221
SL02-SL03
-
+
o
23
22
1
T1=4T(3)/3
NT=1
(Output) OUT 23
Return to 221
NT1=0
(Output) K = 4
Return to 221


6
SLO4=Slope
T7=T(4)-T(2)
T7
-
+
o
7
8
9
DIFF =
SLO4
-SLO5
NT2 = 1
Return
to 221
DIFF =
SLO4
-SLO5
25
-
DIFF
+
o
DIFF
-
27
+
o
SLO1
=SLO4
T(1)=T(4)
20
26
SLO3
=SLO4
T(3)=T(4)
(Output)
OUT 25
(Output)
OUT 27
1
DIFF-EP1
-
o
14
11
-
o
DIFF-EP1
1
+
+
15
12
SLO3=SLO2
T(3)=T(2)
SLO2=SLO4
T(2)=T(4)
SLO5=SLO4
SLO1=SLO2
T(1)=T(2)
SLO2=SLO4
T(2)=T(4)
SLO5=SLO4
(Output)
OUT 15
(Output)
OUT 12
1
1
NT2=1
NT2=1
Return
to 221
Return
to 221

```
      SUBROUTINE SLOMAX
      DIMENSION LTITLE(10),RR(128),RI(128),T(10),SLO(10)
      COMMON RR,RI,T,SLO,G,B,A,PI,SLOPE,NT,NT1,NT2,EP1,I1,T1
      COMMON NT3,TM
      IF(SLOPE)41,42,43
   41 WRITE OUTPUT TAPE 4,44
   44 FORMAT(20H SLOPE 2 IS NEGATIVE/)
      GO TO 43
   42 WRITE OUTPUT TAPE 4,45
   45 FORMAT(26H CALCULATED SLOPE IS ZERO./)
   43 IF(NT1)5,6,6
    5 IF(NT)2,3,4
    2 K=1
      T(K)=T1
      SLO1=SLOPE
      SLO(K)=SLOPE
      IF(NT3)39,38,39
   38 T1=1.*T(2)/2.
      GO TO 40
   39 T1=(B+T1)/2.
   40 K=K+1
      NT=0
      RETURN
    3 T(K)=T1
      SLO2=SLOPE
      SLO(K)=SLOPE
      SLO5=SLO2
      IF(SLO2-SLO1)19,21,21
   19 T1=1.*T(1)/4.
      NT=-1
      NT3=0
      PRINT 35
   35 FORMAT(7H OUT 19/)
      RETURN
   21 T1=TM
      K=K+1
      NT=1
      RETURN
    4 T(K)=T1
      SLO3=SLOPE
      IF(SLO2-SLO3)23,22,22
   23 T1=4.*T(3)/3.
      NT=1
      PRINT 36
   36 FORMAT(7H OUT 23/)
      RETURN
   22 SLO(K)=SLOPE
      K=K+1
    1 C1=SLO1*(T(2)*T(2)-T(3)*T(3))
      C2=-SLO2*(T(1)*T(1)-T(3)*T(3))
      C3=SLO3*(T(1)*T(1)-T(2)*T(2))
      C4=SLO1*(T(2)-T(3))
      C5=-SLO2*(T(1)-T(3))
      C6=SLO3*(T(1)-T(2))
      T1=.5*(C1+C2+C3)/(C4+C5+C6)
      NT1=0
      PRINT 30,K
   30 FORMAT(3H K=I3/)
```

```
      RETURN
    6 ER1=0.0
      SLO4=SLOPE
      T(K)=T1
      SLO(K)=SLOPE
      T7=T(4)-T(2)
      IF(T7)7,8,9
    7 DIFF=SLO4-SLO5
      IF(DIFF)25,20,20
   25 SLO1=SLO4
      SLO(1)=SLO4
      T(1)=T(4)
      PRINT 31
   31 FORMAT(7H OUT 25/)
      GO TO 1
   20 IF(DIFF-EP1)14,14,15
    8 CONTINUE
      NT2=1
      RETURN
    9 DIFF=SLO4-SLO5
      IF(DIFF)27,26,26
   27 SLO3=SLO4
      SLO(3)=SLO4
      T(3)=T(4)
      PRINT 32
   32 FORMAT(7H OUT 27/)
      GO TO 1
   26 IF(DIFF-EP1)11,11,12
   11 CONTINUE
      NT2=1
      RETURN
   12 CONTINUE
      SLO1=SLO2
      SLO(1)=SLO2
      T(1)=T(2)
      SLO2=SLO4
      SLO(2)=SLO4
      T(2)=T(4)
      SLO5=SLO4
      PRINT 33
   33 FORMAT(7H OUT 12/)
      GO TO 1
   14 CONTINUE
      NT2=1
      RETURN
   15 CONTINUE
      SLO3=SLO2
      SLO(3)=SLO2
      T(3)=T(2)
      SLO2=SLO4
      SLO(2)=SLO4
      T(2)=T(4)
      SLO5=SLO4
      PRINT 34
   34 FORMAT(7H OUT 15/)
      GO TO 1
      END
```

```
      SUBROUTINE AVER(S,A,I1,J1,DIF,SUM)
      DIMENSION S(200),A(12,1C)
      I2=I1+1
      N1=J1-I1-1
      DO 1 I=1,I2
       A(I,1)=0.5*(S(N1)+S(N1+1))
    1 N1=N1+1
      I3=I1-1
      N1=I2
      DO 2 I=1,I3
      N1=N1-1
      DO 2 J=1,N1
    2 A(J,I+1)=0.5*(A(J+1,I)+A(J,I))
      DIF=ABSF((A(N1,I3+1)-A(N1-1,I3+1))/A(N1,I3+1))
      SUM=A(N1,I3+1)
      END
      SUBROUTINE GAUSSQ(D,E,SUM)
      DIMENSION XSI(24),WSI(24),X(9),W(9),RR(128),RI(128),T(10),SLO(10)
      COMMCN RR,RI,T,SLO,G,B,A,PI,SLOPE,NT,NT1,NT2,EP1,I1,T1
      COMMON NT3,TM
      IF(W)5,1,5
    1 W=1.0
      XSI(16)=0.1834346425
      XSI(17)=0.5255324099
      XSI(18)=0.7966664774
      XSI(19)=0.9602898565
      XSI(20)=0.0
      XSI(21)=0.3242534234
      XSI(22)=0.6133714327
      XSI(23)=0.8360311073
      XSI(24)=0.9681602395
      WSI(16)=0.3626837834
      WSI(17)=0.3137066459
      WSI(18)=0.2223810345
      WSI(19)=0.1012285363
      WSI(20)=0.3302393550
      WSI(21)=0.3123470770
      WSI(22)=0.2606106964
      WSI(23)=0.1806481607
      WSI(24)=0.0812743884
    5 N1=8
      I2=19
      I3=N1/2
      K2=I2
      A1=(E+D)*0.5
      A2=(E-D)*0.5
   33 I5=N1
      DO 35 I=1,I3
      X(I)=A1-A2*XSI(K2)
      W(I)=A2*WSI(K2)
      X(I5)=A1+A2*XSI(K2)
      W(I5)=W(I)
      K2=K2-1
   35 I5=I5-1
   49 SUM=0.0
      DO 37 I=1,N1
      X1=X(I)
      CALL EVAL35(X1,VR,VI)
   37 SUM=SUM+VI*W(I)
      RETURN
      END
```

```
      SUBROUTINE SCOTS2(R,X,Y)
      DIMENSION R(129),X(129),RR(128),RI(128),IT(10)
      COMMCN G,B,A,T,TM,TD,EP,L1,LD,I1,M1,RR,RI,PI
      R(1)=1.0
      R(2)=B
      R(3)=-(B/A)*(G+1.)
      R(4)=-((B**2)/A)*G
      X(1)=0.0
      X(2)=0.0
      X(3)=-(B/A)*Y
      X(4)=-((B**2)/A)*Y
      NO=3
      MM=0
      IT=0
      CALL TOOTS2(R,X,NO,IT,MM,-.5,+.5)
      IF(RR(1))4,4,5
      RRR=0.
      RII=0.
    5 RRR=RR(1)
      RII=RI(1)
      RR(1)=RR(2)
      RI(1)=RI(2)
      RR(2)=RR(3)
      RI(2)=RI(3)
      RR(3)=RRR
      RI(3)=RII
      GO TO 3
    4 IF(RR(2))3,3,6
    6 RRR=RR(2)
      RII=RI(2)
      RR(2)=RR(3)
      RI(2)=RI(3)
      RR(3)=RRR
      RI(3)=RII
    3 RETURN
      END
```

For **use** with SL02 and SL04.

```
      SUBROUTINE EVAL2S(RR,RI,G,B,Y,C1,VR,VI)
      DIMENSION Z(3),SR(4,4),SI(4,4),E(3),S(3),C(3),RR(128),RI(128),W(3)
      DO 100 I=1,3
      Z(I)=((RR(I)*RR(I)-RI(I)*RI(I))/B)+RR(I)
  100 W(I)=(2.*RR(I)*RI(I)/B)+RI(I)
      DO101 I=1,2
      DO102 J=2,3
      SR(I,J)=(Z(I)*Z(J))-(W(I)*W(J))
  102 SI(I,J)=(Z(I)*W(J))+(Z(J)*W(I))
  101 CONTINUE
      E31=EXPF(RR(3)+RR(1))
      S31=SINF(RI(3)+RI(1))
      C31=COSF(RI(3)+RI(1))
      AR1=E31*C31
      AI1=E31*S31
      E3=EXPF(RR(3))
      S3=SINF(RI(3))
      C3=COSF(RI(3))
      DR1=E3*C3
      DI1=E3*S3
      E1=EXPF(RR(1))
      S1=SINF(RI(1))
      C11=COSF(RI(1))
      ER1=E1*C11
      EI1=E1*S1
      SRN=SR(2,3)-SR(1,2)
      SIN=SI(2,3)-SI(1,2)
      SRD1=SR(2,3)-SR(1,3)
      SID1=SI(2,3)-SI(1,3)
      SRD2=SR(1,3)-SR(1,2)
      SID2=SI(1,3)-SI(1,2)
      ANR=(SRN*AR1)-(SIN*AI1)
      ANI=(SIN*AR1)+(SRN*AI1)
      DNR1=SRD1*DR1-SID1*DI1
      DNI1=SID1*DR1+SRD1*DI1
      DNR2=SRD2*ER1-SID2*EI1
      DNI2=SID2*ER1+SRD2*EI1
      DNR=DNR1+DNR2
      DNI=DNI1+DNI2
      ANUR=ANR*DNR+ANI*DNI
      ANUI=ANI*DNR-ANR*DNI
      DENOM=DNR*DNR+DNI*DNI
      VR=C1*(ANUR/DENOM)
      VI=C1*(ANUI/DENOM)
      END
```

```
      SUBROUTINE EVAL2L(RR,RI,G,B,Y,C1,VR,VI)
      DIMENSION Z(3),SR(4,4),SI(4,4),E(3),S(3),C(3),RR(128),RI(128),W(3)
      DO100 I=1,3
      Z(I)=((RR(I)**2-RI(I)**2)/B)+RR(I)
100   W(I)=(2.*RR(I)*RI(I)/B)+RI(I)
      DO101 I=1,2
      DO102 J=2,3
      SR(I,J)=(Z(I)*Z(J))-(W(I)*W(J))
102   SI(I,J)=(Z(I)*W(J))+(Z(J)*W(I))
101   CONTINUE
      E31=EXPF(RR(3)+RR(1))
      S31=SINF(RI(3)+RI(1))
      C31=COSF(RI(3)+RI(1))
      E32=EXPF(RR(3)+RR(2))
      S32=SINF(RI(3)+RI(2))
      C32=COSF(RI(3)+RI(2))
      E21=EXPF(RR(2)+RR(1))
      C21=COSF(RI(2)+RI(1))
      S21=SINF(RI(2)+RI(1))
      DO 103 I=1,3
      E(I)=EXPF(RR(I))
      S(I)=SINF(RI(I))
103   C(I)=COSF(RI(I))
      AR1=E31*C31
      AI1=E31*S31
      AR2=E21*C21
      AI2=E21*S21
      BR1=AR2
      BI1=AI2
      BR2=E32*C32
      BI2=E32*S32
      CR1=BR2
      CI1=BI2
      CR2=AR1
      CI2=AI1
      AR12=AR1-AR2
      AI12=AI1-AI2
      A1R=(SR(2,3)*AR12)-(SI(2,3)*AI12)
      A1I=(SI(2,3)*AR12)+(SR(2,3)*AI12)
      DR1=E(3)*C(3)
      DI1=E(3)*S(3)
      DR2=E(2)*C(2)
      DI2=E(2)*S(2)
      DR12=DR1-DR2
      DI12=DI1-DI2
      ER1=E(1)*C(1)
      EI1=E(1)*S(1)
      ER2=DR1
      EI2=DI1
      ER12=ER1-ER2
      EI12=EI1-EI2
      FR1=DR2
      FI1=DI2
      FR2=ER1
      FI2=EI1
      FR12=FR1-FR2
      FI12=FI1-FI2
      BR12=BR1-BR2
      BI12=BI1-BI2
      CR12=CR1-CR2
      CI12=CI1-CI2
      B2R=(SR(1,3)*BR12)-(SI(1,3)*BI12)
      B2I=(SI(1,3)*BR12)+(SR(1,3)*BI12)
      C3R=(SR(1,2)*CR12)-(SI(1,2)*CI12)
      C3I=(SI(1,2)*CR12)+(SR(1,2)*CI12)
      D1R=(SR(2,3)*DR12)-(SI(2,3)*DI12)
      D1I=(SI(2,3)*DR12)+(SR(2,3)*DI12)
      E2R=(SR(1,3)*ER12)-(SI(1,3)*EI12)
      E2I=(SI(1,3)*ER12)+(SR(1,3)*EI12)
      F3R=(SR(1,2)*FR12)-(SI(1,2)*FI12)
      F3I=(SI(1,2)*FR12)+(SR(1,2)*FI12)
      ANR=A1R+B2R+C3R
      ANI=A1I+B2I+C3I
      DNR=D1R+E2R+F3R
      DNI=D1I+E2I+F3I
      ANUR=ANR*DNR+ANI*DNI
      ANUR=ANR*DNR+ANI*DNI
      ANUI=ANI*DNR-ANR*DNI
      ANUI=ANI*DNR-ANR*DNI
      DENOM=DNR*DNR+DNI*DNI
      DENOM=DNR*DNR+DNI*DNI
      VR=C1*(ANUR/DENOM)
      VI=C1*(ANUI/DENOM)
      END
```

For use with SL02 and SL04

when NTU less than 5

For use with TEMP2 and TEMP4

```
      SUBROUTINE EVAL53S(X1,VR,VI)
      DIMENSION Z(3),SR(4,4),SI(4,4),E(3),S(3),C(3),RR(128),RI(128),W(3)
     1,R(129),X(129)
      COMMON G,B,A,T,TM,TD,EP,L1,LD,I1,N1,RR,RI,PI
      Y=(PI/T)*X1
      CALL SOOTS2(R,X,Y)
      C1=1.0
      DO 100 I=1,3
      Z(I)=((RR(I)*RR(I)-RI(I)*RI(I))/B)+RR(I)
100   W(I)=(2.*RR(I)*RI(I)/B)+RI(I)
      DO101 I=1,2
      DO102 J=2,3
      SR(I,J)=(Z(I)*Z(J))-(W(I)*W(J))
102   SI(I,J)=(Z(I)*W(J))+(Z(J)*W(I))
101   CONTINUE
      E31=EXPF(RR(3)+RR(1))
      S31=SINF(RI(3)+RI(1))
      C31=COSF(RI(3)+RI(1))
      AR1=E31*C31
      AI1=E31*S31
      E3=EXPF(RR(3))
      S3=SINF(RI(3))
      C3=COSF(RI(3))
      DR1=E3*C3
      DI1=E3*S3
      E1=EXPF(RR(1))
      S1=SINF(RI(1))
      C11=COSF(RI(1))
      ER1=E1*C11
      EI1=E1*S1
      SRN=SR(2,3)-SR(1,2)
      SIN=SI(2,3)-SI(1,2)
      SRD1=SR(2,3)-SR(1,3)
      SID1=SI(2,3)-SI(1,3)
      SRD2=SR(1,3)-SR(1,2)
      SID2=SI(1,3)-SI(1,2)
      ANR=(SRN*AR1)-(SIN*AI1)
      ANI=(SIN*AR1)+(SRN*AI1)
      DNR1=SRD1*DR1-SID1*DI1
      DNI1=SID1*DR1+SRD1*DI1
      DNR2=SRD2*ER1-SID2*EI1
      DNI2=SID2*ER1+SRD2*EI1
      DNR=DNR1+DNR2
      DNI=DNI1+DNI2
      DENR=(G*DNR)-Y*DNI
      DENI=(Y*DNR)+G*DNI
      DENOM=(DENR*DENR)+DENI*DENI
      VR=C1*COSF(PI*X1)*(ANR*DENR+ANI*DENI)/DENOM
      VI=-C1*SINF(PI*X1)*(ANI*DENR-ANR*DENI)/DENOM
      END
```

For use with SLO3 and SLO5, $\lambda = 0$.

```
 SUBROUTINE EVAL35(X1,VR,VI)
 DIMENSION VR(20),VI(20),SU(200),AV(12,10),R(129),X(129),ER(200),
1RR(128),RI(128),T(10),SLO(10)
 COMMON RR,RI,T,SLO,G,B,A,PI,SLOPE,NT,NT1,NT2,EP1,I1,T1
 COMMON NT3,TM
 Y=(PI/T1)*X1
 C1=1.
 D1=((G*G)+G+(Y*Y))/((G+1.)*(G+1.)+(Y*Y))
 E1=EXPF(-B*D1)
 F=Y/((G+1.)*(G+1.)+(Y*Y))
 C=COSF(B*F)
 S=SINF(B*F)
 VR=C1*E1*C*COSF(PI*X1)
 VI=C1*E1*S*SINF(PI*X1)
 END
 END
```

For use with TEMP2 and TEMP4, $\lambda = 0$

```
 SUBROUTINE EVAL3T(X1,VR,VI)
 DIMENSION Z(3),SR(4,4),SI(4,4),E(3),S(3),C(3),RR(128),RI(128),W(3)
1,R(129),X(129)
 COMMON G,B,A,T,TM,TD,EP,L1,LD,I1,M1,RR,RI,PI
 Y=(PI/T)*X1
 C1=1.0
 G1=G+1.
 DEN=G1*G1+Y*Y
 D1=(G*G+Y*Y)/DEN
 F1=(G1*Y-G*Y)/DEN
 E1=EXPF(-B*D1)
 C=COSF(B*F1)
 S=SINF(B*F1)
 DENOM=G*G+Y*Y
 VR=C1*COSF(PI*X1)*E1*(C*G+S*Y)/DENOM
 VI=C1*SINF(PI*X1)*E1*(S*G-C*Y)/DENOM
 END
```